

Information and Communication Technology (A Technical Subject)

Commenced from 2006 January

for

G.C.E. (O/L)

Resource Material for Teachers

(Unit Four to Eight)



IT Department
National Institute of
Education
Maharagama
Sri Lanka



Secondary Education
Modernization Project
Ministry of Education
Funded by the Asian
Development Bank

Table of contents

4 Information Systems	01
4.1 Introduction to Systems	
4.1.1 System Components	01
4.1.2 Organizations as Systems	02
4.2 Types of Processing Systems	
4.2.1 Batch Processing	03
4.2.2 Online Real time Processing	03
4.3 Types of Information Systems	
4.3.1 Office Automation System	04
4.3.2 Transaction Processing System	04
4.3.3 Management Information System	04
4.3.4 Processing Control System	04
4.3.5 Intelligent Information System	05
4.4 Introduction to System Development Life Cycle	
4.4.1 Analysis	05
4.4.2 Design	06
4.4.3 Implementation and Testing	06
4.4.4 Maintenance	07
5 Programming Concepts	
5.1 Introduction to Programming	
5.1.1 What is a Program	09
5.1.2 Programming Languages (Classification, Generations, Compilers and Interpreters)	10
5.2 Algorithms (Flowcharts and Pseudo Cords)	14
5.3 Data types and Operators	36

5.4 Control Structures	47
5.4.1 Sequence	
5.4.2 Selection	
5.4.3 Repetition	
5.4 Control Structures	
5.5 Data Input / Output	
5.6 Arrays	
5.7 Sub Routines	
5.8 Database handling	
6. Web Site Development	
6.1 Web Page Development	68
6.1.1 Fundamentals of HTML and XML	70
6.1.2 Creating a simple web page	70
6.1.3 Creating and linking multiple pages	71
6.2 Web development Tools	86
6.3 Requirements of web publishing	124
6.3.1 ISP	
6.3.2 Web Server	
6.3.3 URL.	
6.3.4 IP Address.	
6.4 Design multimedia contents for web sites	129
6.4.1 Graphic Designing	
6.4.2 Animation	
6.4.3 Sound Recording	
6.4.4 Incorporating Audio and Video Animations	

7. ICT and society	141
7.1 ICT for National Development.	141
7.1.1 Health	142
7.1.2 Education	144
7.1.3 Agriculture	145
7.1.4 Industries	145
7.1.5 Other Services	147
7.2 Issues in the use of ICT.	150
7.2.1 Ethical	150
7.2.2 Legal	150
7.2.3 Security	151
7.2.4 Health and Safety	155
7.2.5 Social	158
8. Group Project	160

Advisers

Prof. J.W.Wickramasinghe
Director General
National Institute of Education
Maharagama

Dr. Mrs. I.L.Ginige
Asst. Director General
National Institute of Education
Maharagama

Mr. H.Jayasinghe
Director
IT Department
National Institute of Education
Maharagama

Resource Team

Sydney Jayawardene
Chief Project Officer
IT Department – NIE

A.M.Kanthi
Project Officer
IT Department – NIE

Suriyaarachchi S.K.N.
Lecturer
Provincial ICT Education
Center
WP, Pannipitiya

Ruwan Abesekera
Road
Lecturer
Provincial ICT Education
Center
WP, Pannipitiya

Education
Wijsekara W.M.A.S
Centre Manager
Haliela CRC

Padmasiri W.E.M.S
Lecturer
Provincial ICT Education
Center
Karunanayake
NWP, Wariyapola

Ramanayaka D.S.B
Lecturer
Provincial ICT Education
Center
NWP, Wariyapola

Ranjan R.P
Lecturer
Provincial ICT Education
Center
SP, Galle

Kumarihami D.M.D
ISA
Zonal Ed Office-Welimada

Rathnayaka L.K
ISA
Zonal Ed Office-Passara

Aluthwala N
Lecturer
Provincial ICT Education
Center
Sabaragamuwa

Samarasinghe M.B
Lecturer

Provincial ICT Education
Center

Vazeer A.M
Centre Manager
Kahagolla CRC

NobelRavi N
Center Manager
Nuwaraeliya CRC

Jayarathna Banda A.N
Lecturer
Gurudeniya CRC

Percival K.D.D
Retired Lecturer
Prethibimbarama

Dehiwala

Bandara R.M.P
Lecturer
Provincial ICT

Center
CP, Gurudeniya

Palitha Wimalaweera
Instructor
Thopawewa CRC

Niranjan

Instructor
Galahitiyawa CRC

Unit 4. Information Systems

4.1 Introduction to systems.

What is a system?

A system is a collection of interrelated components that work together to perform a specific task or achieve a goal. In a system, the different components are connected with each other and they are interdependent.

4.1.1 System Composition

What is a subsystem?

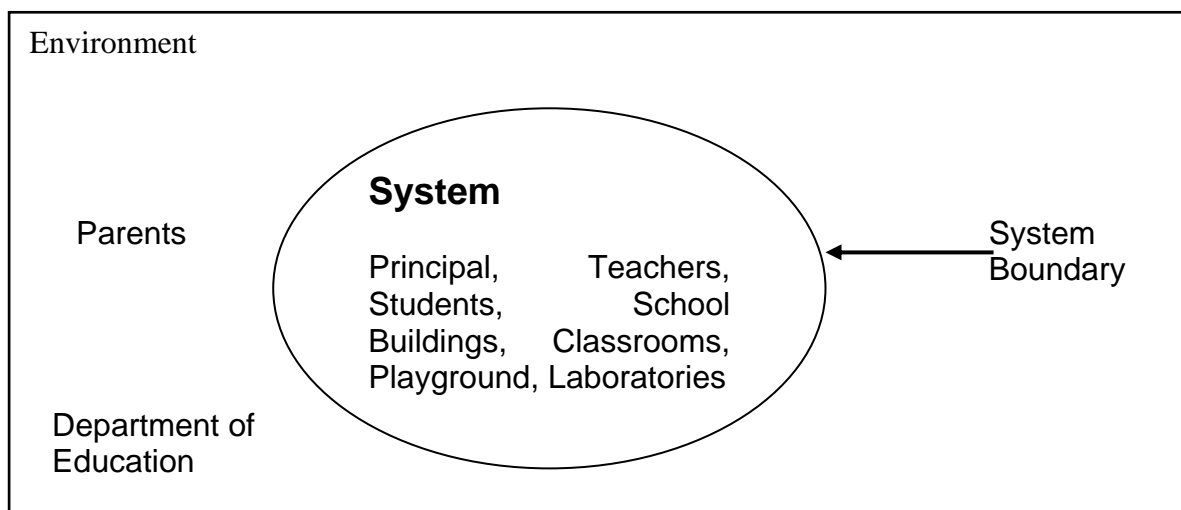
Each component is a subsystem of the main (original) system and carries out a part of the system task.

For example, the human body represents a complete natural system. The human body contains complex muscle, bone, respiratory, digestive and circulatory subsystems, each providing a specific part of the system task. School, Bank, Library, Supermarket, are some other examples of systems.

What is system boundary?

A system usually interacts with the external world or environment. The system boundary separates the system from its environment. A system's boundary tells us what is inside and what is outside the system.

Let us consider the school for example and identify the system boundary.

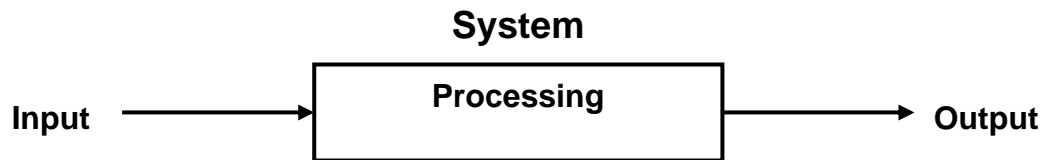


in the above example , the system boundary clearly illustrates the components inside the school and components outside the system.

What are the elements of a system?

Input
Process
Output

To perform a task or accomplish a goal, a system receives inputs (from the environment), process and returns outputs (to the environment).



If we consider a school as a system,

Goal : Education of students
Input : Children, Teachers, Funds
Processing : Teaching and learning
Output : Educated students

4.1.2 Organizations as Systems.

Can we consider an organization as a System?

If we look in our surroundings, we see many organizations: such as shops, medical centers, communication centers, theatres, universities and others. All of these organizations have **goals, components, inputs, processing** and **outputs**. They can be viewed as systems and analyzed to identify system boundaries, system types and the environment in which they operate.

What is Information System?

Data are raw facts. Information is processed data. Information is used to take decisions. We use a process to turn data into information. A process is a set of logically related tasks performed to achieve a specific outcome. Sometimes, data are processed mentally or manually.



What is Manual Information system?

In a manual Information system, all data processing is done manually. Filing cabinets, manual procedures and papers are vital components of a manual information system.

Example

- Book- keeping system
- School registration and attendance keeping system
- Library books index system

What is Computer-based Information system?

A computer-based information system is a single set of hardware, software, databases, telecommunications, people and procedures that are configured to collect, process, manipulate, store, and communicate data into information.

Example

- Telephone billing system
- Results produced by the department of examinations

4.2 Types of Processing Systems.

What are the types of processing Systems

Batch processing
On line Real time Processing

4.2.1 Batch processing?

Data is collected for a period of time and processed batch wise. The term originated in the days when users entered programs on punch cards. These cards are fed into the computer batch wise.

4.2.2 On line Real time processing?

A type of computer processing is one in which the computer responds immediately to user requests. Each request is considered to be a transaction. Automatic teller machines in banks are an example of transaction processing.

The opposite of transaction processing is batch processing, in which a batch of requests is stored and then executed all at one time. Transaction processing requires interaction with a user, whereas batch processing can take place without a user being present.

4.3 Types of Information Systems.

- Office Automation System
- Transaction processing system
- Management information System
- Processing control System

4.3.1 Office automation System.

Office Automation systems (**OAS**) are systems that try to improve the productivity of employees who need to process data and Information.

Example :

Microsoft Office Package

4.3.2 Transaction processing systems.

Transaction processing systems are information system applications that capture and process data about (or for) business transactions. They are sometimes called Data Processing Systems (DPS). A TPS is focused at the operational level of a business.

The management information produced by transaction processing systems usually consists of detail reports of daily transactions or future transactions. The reports generated by a TPS are useful only to lower - level managers.

A TPS usually operates only within one functional area of a business. For example, in an organization, each department, marketing, accounting and finance, production, and research and development has its own TPS.

The main components of a Transaction Processing System are hardware (machines), software (instructions or programs), people (programmers, managers, or users), procedures (rules), data and information.

What are the differences between manual System and automated TPS

An automated system is more Efficient, accurate and more reliable. Automated TPS use different components with compared to the components of a manual system

4.3.3 Management Information System.

A management Information System (MIS) is mainly concerned with the internal sources of information. MIS usually take data from the transaction processing systems and summarize it into a series of management reports.

MIS reports tend to be used by middle management and operational supervisors.

4.3.4 Processing control systems

Process control systems are used to control machines and equipment. e.g.: cement production factory, vehicle manufacturing and assembling , robots etc.

4.3.5 Intelligent information system

A computer application that does the work of human experts. For example, there are expert systems that can diagnose human illnesses, make financial forecasts, and schedule routes for delivery vehicles. Some expert systems are designed to take the place of human experts, while others are designed to aid them.

Expert systems are part of a general category of computer applications known as artificial intelligence . To design an expert system, one needs a knowledge engineer, an individual who studies how human experts make decisions and translates the rules into terms that a computer can understand.

4.4 Introduction to System development life Cycle.

System Development Life Cycle (SDLC) is the overall process of developing any kind of systems. It has several steps..

Steps of System Development Life Cycle

- Identifying the problem (user requirement)
- Feasibility Study
- System Analysis
- System Design
- Software Development
- Testing
- Implementation
- Maintenance

4.4.1 Identifying the problem (user requirement)

The initiation of a system begins when a business need or opportunity is identified. This is called identifying the problem. A Project Manager should be appointed to manage the project. This business need is documented in a Concept Proposal. After the Concept Proposal is approved, the System Development Phase begins.

4.4.2 Feasibility Study

Phase Two of the SDLC is a Feasibility study. If it is known what the system is to accomplish, now it needs to be decided if the system CAN be built feasibly. Feasibility is based on three main factors.

1. Do we have the technology to construct the system we want?
2. Is the project fiscally possible (i.e. can the purchaser afford it)?
3. Is it practical for the system to be built?

Do we have the technology to construct it? This question is almost a misnomer, it is rare to find a system that we cannot construct with the technical level that we currently have ... but there will be times when technology will be a limiting factor. More than likely, however, the technology will be there, it is just a matter of deciding if the cost can be justified...

Fiscal feasibility. Is the gain(s) made by the new system worth the cost of producing the new system? How much does it cost to maintain the current system (actual maintenance costs, lost revenue due to poor service, loss of productivity caused by waiting for the old system, etc.)

4.4.3 System Analysis

In this phase, the systems analysts study the existing system in detail, and identify new requirements. Analysis gathers the requirements for the system. This stage includes a detailed study of the business needs of the organization. Options for changing the business process may be considered

4.4.4 System design

Design focuses on high level design like, what programs are needed and how they are going to interact, low-level design (how the individual programs are going to work), interface design (what are the interfaces going to look like) and data design (what data will be required). During these phases, the software's overall structure is defined. Analysis and Design are very crucial in the whole development cycle. Any glitch in the design phase could be very expensive to solve in the later stage of the software development. Much care is taken during this phase. The logical system of the product is developed in this phase

4.4.5 Software Development

In this phase the designs are translated into code. Computer programs are written using a conventional programming language or an application generator. Programming tools like Compilers, Interpreters, Debuggers are used to generate the code. Different high level programming languages like C, C++, Pascal, Java are used for coding. With respect to the type of application, the right programming language is chosen (The design specifications are converted into source code of a programming language.)

4.4.6 Testing

In this phase the system is tested. Normally programs are written as a series of individual modules, these are subject to separate and detailed test. The system is then tested as a whole. The separate modules are brought together and tested as a complete system. The system is tested to ensure that interfaces between modules work (integration testing), the system works on the intended platform and with the expected volume of data (volume testing) and that the system does what the user requires (acceptance/beta testing).

4.4.7 Implementation

The implementation stage of any project is a true display of the defining moments that make a project a success or a failure. The implementation stage is defined as "the system or system modifications being installed and made operational in a production environment. The phase is initiated after the system has been tested and accepted by the user. This phase continues until the system is operating in production in accordance with the defined user requirements", while all of the planning that takes place in preparation of the implementation phase is critical and important.

4.4.7 Maintenance

Inevitably the system will need maintenance. Software will definitely undergo change once it is delivered to the customer. There are many reasons for the change. Change could happen because of some unexpected input values into the system. In addition, the changes in the system could directly affect the software operations. The software should be developed to accommodate changes that could happen during the post implementation period.

Some e-books and web sites give various definitions and explanations about phases of SDLC. Following is also another description about phases of SDLC and they can be used as ***optional*** steps.

Initiation Phase

The initiation of a system (or project) begins when a business need or opportunity is identified. A Project Manager should be appointed to manage the project. This business need is documented in a Concept Proposal. After the Concept Proposal is approved, the System Concept Development Phase begins.

System Concept Development Phase

Once a business need is approved, the approaches for accomplishing the concept are reviewed for feasibility and appropriateness. The Systems Boundary Document identifies the scope of the system and requires Senior Official approval and funding before beginning the Planning Phase.

Planning Phase

The concept is further developed to describe how the business will operate once the approved system is implemented, and to assess how the system will impact employee and customer privacy. To ensure the products and /or services provide the required capability on-time and within budget, project resources, activities, schedules, tools, and reviews are defined. Additionally, security certification and accreditation activities begin with the identification of system security requirements and the completion of a high-level vulnerability assessment.

Requirements Analysis Phase

Functional user requirements are formally defined and delineate the requirements in terms of data, system performance, security, and maintainability requirements for the system. All requirements are defined to a level of detail sufficient for systems design to proceed. All requirements need to be measurable and testable and relate to the business need or opportunity identified in the Initiation Phase.

Design Phase

The physical characteristics of the system are designed during this phase. The operating environment is established, major subsystems and their inputs and outputs are defined, and processes are allocated to resources. Everything requiring user input or approval must be documented and reviewed by the user. The physical characteristics of the system are specified and a detailed design is prepared. Subsystems identified during design are used to create a detailed structure of the system. Each subsystem is partitioned into one or more design units or modules. Detailed logic specifications are prepared for each software module.

Development Phase

The detailed specifications produced during the design phase are translated into hardware, communications, and executable software. Software shall be unit tested, integrated, and retested in a systematic manner. Hardware is assembled and tested.

Integration and Test Phase

The various components of the system are integrated and systematically tested. The user tests the system to ensure that the functional requirements, as defined in the functional requirements document, are satisfied by the developed or modified system. Prior to installing and operating the system in a production environment, the system must undergo certification and accreditation activities.

Implementation Phase

The system or system modifications are installed and made operational in a production environment. The phase is initiated after the system has been tested and accepted by the user. This phase continues until the system is operating in production in accordance with the defined user requirements.

Operations and Maintenance Phase

The system operation is ongoing. The system is monitored for continued performance in accordance with user requirements, and needed system modifications are incorporated. The operational system is periodically assessed through In-Process Reviews to determine how the system can be made more efficient and effective. Operations continue as long as the system can be effectively adapted to respond to an organization's needs. When modifications or changes are identified as necessary, the system may reenter the planning phase.

Disposition Phase

The disposition activities ensure the orderly termination of the system and preserve the vital information about the system so that some or all of the information may be reactivated in the future if necessary. Particular emphasis is given to proper preservation of the data processed by the system, so that the data is effectively migrated to another system or archived in accordance with applicable records management regulations and policies, for potential future access

Unit 5

Programming Concepts

Computer programming (often simply programming or coding) is the craft of writing a set of commands or instructions that can later be compiled and/or interpreted and then inherently transformed to an executable that an electronic machine can execute or "run".

Programming requires mainly logic, but has elements of science, mathematics, engineering, and many would argue it as an art.

In software engineering, programming is regarded as one phase in a software development process.

A first-generation programming language is a machine-level programming language. It consists of 1s and 0s.

Originally, no translator was used to compile or assemble the first-generation language. The first-generation programming instructions were entered through the front panel switches of the computer system.

The main benefit of programming in a first-generation programming language is that code a user writes can run very fast and efficiently since it is directly executed by the CPU, but machine language is somewhat more difficult to learn than higher generational programming languages, and it is far more difficult to edit if errors occur, or, for example, if instructions need to be added into memory at some location, then all the instructions after the insertion point need to be moved down to make room in the memory to accommodate the new instructions. Doing so on a front panel with switches can be very difficult. Furthermore portability is significantly reduced - in order to transfer code to a different computer it needs to be completely rewritten since the machine language for one computer could be significantly different from another computer. Architectural considerations make portability difficult too. For example, the number of registers on one CPU architecture could differ from those of another.

A **second-generation programming language** is a term usually used to refer to some form of assembly language. Unlike first-generation programming languages, the code can be read and written fairly easily by a human, but it must be converted into a machine readable form in order to run on a computer. The conversion process is simply a mapping of the assembly language code into binary machine code (the first-generation language). The language is specific to and dependent on a particular processor family and environment. Since it is the native language of a processor it has significant speed advantages, but it requires more programming effort and is difficult to use effectively for large applications

A **third generation language (3GL)** is a programming language designed to be easier for a human to understand, including things like named variables. Fortran, ALGOL and COBOL are early examples of this sort of language. Most "modern" languages (BASIC, C, C++, Delphi, Java, and including COBOL, Fortran, ALGOL) are third generation. Most 3GLs support structured programming.

A **fourth-generation programming language** (abbreviated **4GL**) is a programming language designed with a specific purpose in mind, such as the development of commercial business software. Such languages arose after the introduction of modern, block-structured

third-generation programming languages, which improved the process of software development. However, it was still considered by some to be frustrating, slow, and error prone to program computers. This led to the first "programming crisis", in which the amount of work that might be assigned to programmers greatly exceeded the amount of programmer time available to do it. Meanwhile, a lot of experience was gathered in certain areas, and it became clear that certain applications could be generalized by adding limited programming languages to them.

A **fifth-generation programming language** (abbreviated **5GL**) is a programming language based around solving problems using constraints given to the program, rather than using an algorithm written by a programmer. Most constraint-based and logic programming languages and some declarative languages are fifth-generation languages.

While fourth-generation programming languages are designed to build specific programs, fifth-generation languages are designed to make the computer solve the problem for you. This way, the programmer only needs to worry about what problems need to be solved and what conditions need to be met, without worrying about how to implement a routine or algorithm to solve them. Fifth-generation languages are used mainly in artificial intelligence research. Prolog, OPS5, and Mercury are the best known fifth-generation languages.

In the 1990s, fifth-generation languages were considered to be the wave of the future, and some predicted that they would replace all other languages for system development, with the exception of low-level languages. Most notably, Japan put much research and money into their fifth-generation computer systems project, hoping to design a massive computer network of machines using these tools.

However, as larger programs were built, the flaws of the approach became more apparent. It turns out that, starting from a set of constraints defining a particular problem, deriving an efficient algorithm to solve it is a very difficult problem in itself. This crucial step cannot yet be automated and still requires the insight of a human programmer.

Today, fifth-generation languages have lost part of their initial appeal and are mostly used in academic circles.

Compiler

A compiler is a computer program (or set of programs) that translates text written in a computer language (the *source language*) into another computer language (the *target language*). The original sequence is usually called the *source code* and the output called *object code*. Commonly the output has a form suitable for processing by other programs (e.g., a linker), but it may be a human readable text file.

The most common reason for wanting to translate source code is to create an executable program. The name "compiler" is primarily used for programs that translate source code from a high level language to a lower level language (e.g., assembly language or machine language). A program that translates from a low level language to a higher level one is a *decompiler*. A program that translates between high-level languages is usually called a *language translator*, *source to source translator*, or *language converter*. A *language rewriter* is usually a program that translates the form of expressions without a change of language.

A compiler is likely to perform many or all of the following operations: lexing, preprocessing, parsing, semantic analysis, code optimizations, and code generation.

Interpreter

An **interpreter** is a computer program that executes other programs. This is in contrast to a compiler which does not execute its input program (the source code) but translates it into another language, usually executable machine code (also called object code) which is output to a file for later execution. It may be possible to execute the same source code either directly by an interpreter or by compiling it and then executing the machine code produced.

Interpreter versus compiler

It takes longer to run a program under an interpreter than to run the compiled code but it can take less time to interpret it than the total time required to compile and run it. This is especially important when prototyping and testing code when an edit-interpret-debug cycle can often be much shorter than an edit-compile-run-debug cycle.

Interpreting code is slower than running the compiled code because the interpreter must analyse each statement in the program each time it is executed and then perform the desired action whereas the compiled code just performs the action. This run-time analysis is known as "interpretive overhead". Access to variables is also slower in an interpreter because the mapping of identifiers to storage locations must be done repeatedly at run-time rather than at compile time.

There are various compromises between the development speed when using an interpreter and the execution speed when using a compiler. Some systems (e.g. some LISPs) allow interpreted and compiled code to call each other and to share variables. This means that once a routine has been tested and debugged under the interpreter it can be compiled and thus benefit from faster execution while other routines are being developed. Many interpreters do not execute the source code as it stands but convert it into some more compact internal form. For example, some BASIC interpreters replace keywords with single byte tokens which can be used to find the instruction in a jump table. An interpreter might well use the same lexical analyzer and parser as the compiler and then interpret the resulting abstract syntax tree.

High-level programming language

A **high-level programming language** is a programming language that, in comparison to low-level programming languages, may be more abstract, easier to use, or more portable across platforms. Such languages often abstract away CPU operations such as memory access models and management of scope.

Features of high-level languages

The term "high-level language" does not imply that the language is always superior to low-level programming languages but rather refers to the higher level of abstraction from machine language. Rather than dealing with registers, memory addresses and call stacks, high-level languages deal with variables, arrays and complex arithmetic or boolean expressions. In addition, they have no opcodes that can directly compile the language into machine code, unlike low-level languages like Assembly language. Other features such as string handling routines, object-oriented language features and file input/output may also be present.

In general, high-level languages make complex programming simpler, while low-level languages tend to produce more efficient code. In a high-level language, complex elements can be broken up into simpler, though still fairly complex, elements for which the language

provides abstractions, keeping programmers from having to "reinvent the wheel." The cost of this convenience is often less efficient code overall. For this reason, code which needs to run particularly quickly and efficiently may be written in a lower-level language, even if a higher-level language would make the coding easier.

Note that the terms "high-level" and "low-level" are inherently relative. Originally, assembly language was considered low-level and COBOL, C, etc. were considered high-level, as they allowed the abstractions of functions, variables and expression evaluation, and also that they had to be compiled to assembly before being compiled into machine code. Many programmers today might refer to C as low-level, as it still allows memory to be accessed by address, and provides direct access to the assembly level. For more on this distinction, see the external link below.

Low-level programming language

In computer science, a **low-level programming language** is a language that provides little or no abstraction from a computer's microprocessor. The word "low" does not imply that the language is inferior to high-level programming languages but rather refers to the small or nonexistent amount of abstraction between the language and machine language; because of this, low-level languages are sometimes described as being "close to the hardware."

"High-level" and "low-level" are also used relatively; a Java programmer might consider C to be a comparatively low-level language.

Low-level programming languages are sometimes divided into two categories: *first generation*, and *second generation*.

First generation

The first-generation programming language, or *1GL*, is machine code. It is the only language a microprocessor can understand natively. Currently, programmers almost never write programs directly in machine code, because not only does it (like assembly language) require attention to numerous details which a high-level language would handle automatically, but it also requires memorizing or looking up numerical codes for every instruction that is used, which in assembly language would be written as something more readable like "ADD CX INTEREST" or "RET".

Second generation

The second-generation programming language, or *2GL*, is assembly language. It is considered a second-generation language because while it is not a microprocessor's native language, an assembly language programmer must still understand the microprocessor's unique architecture (such as its *registers* and *instructions*).

Algorithm

If you need to get a computer to do a particular task you can program it. Writing a computer program involves describing the solution in a computer language. One does not directly start writing a program in the computer. You are required to analyze the problem, design a solution and to test the solution before you come to the coding phase.

Just imagine that your parents are going to construct a new house. They might discuss with you how many bedrooms they are going to have etc, whether it's a two storey house. It is

unlikely that your parents will get a masion and start building the house directly. Just imagine after constructing the house half way through your parents figure out that the living room is too small. Instead they will get an Architect to draw some plans of the house first, they might show you these plans, so you could visualize how the new house would look like. You and your parents could make changes to the house and have the plans redrawn. After every one is happy about the new house, your parents could meet a contractor and start constructing the house.

Programming is very similar, coding directly is usually not a good idea even for small problems. You need to do some planning before hand. There are many ways that you could represent your solution to the problem (algorithm). There are graphical techniques like flow charts, and textual methods like pseudo code. Once you are satisfied that your solution will work you can convert this into a programming language.

Problem Solving

The most difficult part of solving a problem on a computer is discovering the method of solution. Once you have got a proper solution you could easily convert this into a programming language.

It is therefore helpful to temporarily ignore the programming language and Concentrate on formulating the steps of the solution and design them down in a algorithm.

That algorithm may be in **Flow chart** Or **Pseudo code**

A sequence of instructions expressed in this way is frequently referred to as an algorithm. Writing a computer program involves performing the following tasks.

1. Understanding the problem
2. Developing an Algorithm for the problem
3. Writing a Computer Program
4. Testing the Computer program

STEP 1 – Understand the problem

How do you solve that problem ? The first task is to clearly understand what the problem is. Find out the inputs ? Here the input should be two numbers. Find the output be ? Total. So there must be some calculation that has to be done. That is Process.

STEP 2 – Developing an Algorithm for the problem

With the knowledge you have, you can write a Flow chart solution to the above problem.

STEP 3 – Writing the Computer Program in Visual Basic

Next you go to Visual Basic & start thinking about suitable variable names, data types etc at this stage. Without much difficulty you should be now able to convert the Flow Chart into Visual Basic Coding.

STEP 4 – Testing your computer Program

This step means Checking the accuracy of the program. Here the program runs & produces the results. In case of any errors, you should remove them (that is Debugging).

Programming design techniques

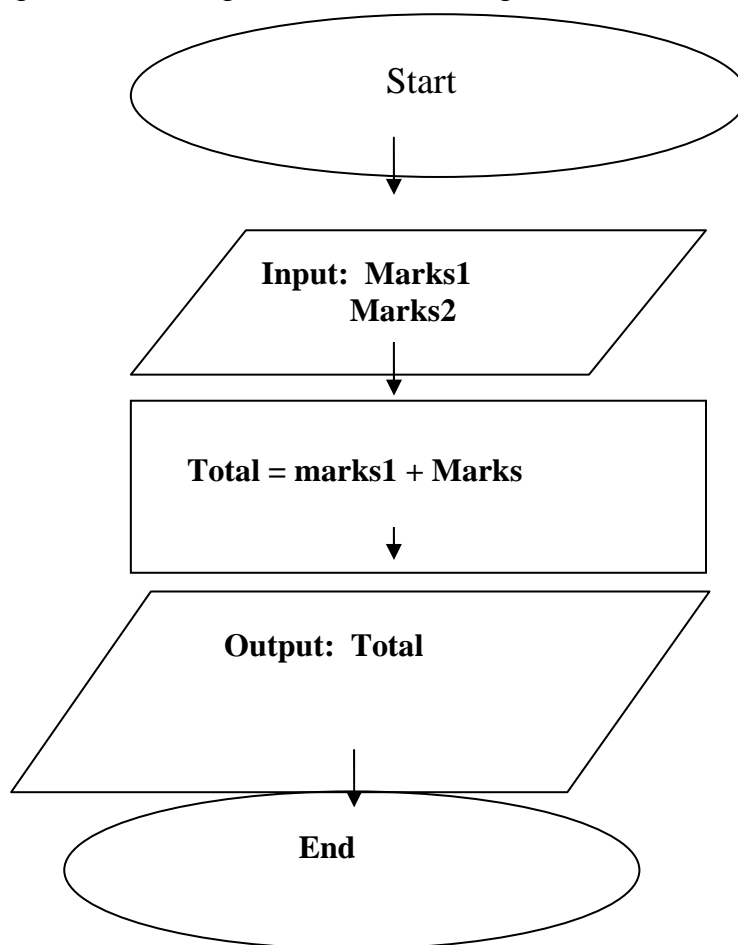
Flowchart & Pseudocode

Flowchart

A **flow chart** is a schematic representation of a process..

Flow-charts can be created by hand or manually in most office software, but lately specialized diagrams drawing software have emerged that can also be used for the purpose, such as Visio, OpenOffice.org Draw, ConceptDraw, Dia, SmartDraw, and OmniGraffle. Programs have been written to create flowcharts directly from the computer program source. Flowcharts may contain other symbols, such as connectors, usually represented as circles, to represent converging paths in the flow chart. Circles will have more than one arrow coming into them but only one going out. Some flow charts may just have an arrow point to another arrow instead. These are useful to represent an iterative process (what in Computer Science is called, a loop). A loop may, for example, consist of a connector where control first enters, processing steps, a conditional with one arrow exiting the loop, and one going back to the connector.

Program for adding two numbers using Flow Chart



Pseudocode

Pseudocode (derived from pseudo and code) is a description of a computer programming algorithm that uses the structural conventions of programming languages, but omits detailed subroutines or language-specific syntax. It can also refer to a high level 'language' whose aim is to generalise the logic and program flow of a computer program

In the context of the Short Code language, pseudocoding refer to the use of codes to represent assembly instructions, even though such codes could not be automatically compiled into an executable program. This usage has mostly fallen out of use.

Program for adding two numbers using Pseudocode

Begin

Input: first number and second number

Total = first Number + Second Number

Output: Total

End

Visual Basic 6.0

1. Introduction to the Visual Basic Language and Environment

Objectives

- Understand the benefits of using Microsoft Visual Basic 6.0 for Windows as an application tool
- Understand the Visual Basic event-driven programming concepts, terminology, and available tools
- Learn the fundamentals of designing, implementing, and distribution of a Visual Basic application
- Learn to use the Visual Basic toolbox
- Learn to modify object properties
- Learn object methods
- Use the menu design window
- Understand proper debugging and error-handling procedures
- Gain a basic understanding of database access and management using databound controls
- Obtain an introduction to ActiveX controls and the Windows Application Programming Interface (API)

What is Visual Basic?

- **Visual Basic** is a tool that allows you to develop Windows (Graphic User Interface - **GUI**) applications. The applications have a familiar appearance to the user.
- Visual Basic is **event-driven**, meaning code remains idle until called upon to respond to some event (button pressing, menu selection, ...). Visual Basic is governed by an event processor. Nothing happens until an event is detected. Once an event is detected, the code corresponding to that event (event procedure) is executed. Program control is then returned to the event processor.

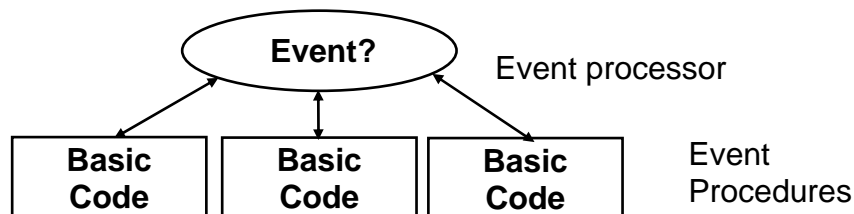
A Brief History of Basic

Language developed in early 1960's at Dartmouth College:

B (eginner's)
A (All-Purpose)
S (Symbolic)
I (Instruction)
C (Code)

- Answer to complicated programming languages (FORTRAN, Algol, Cobol ...). First timeshare language.
- In the mid-1970's, two college students wrote the first Basic for a microcomputer (Altair) that cost \$350 on cassette tape. You may have heard of them: Bill Gates and Paul Allen!
- Every Basic since then is essentially based on that early version. Examples include: GW-Basic, QBasic, QuickBasic.

Visual Basic was introduced in 1991.



- Some Features of Visual Basic
 - Full set of objects - you 'draw' the application
 - Lots of icons and pictures for your use
 - Response to mouse and keyboard actions
 - Clipboard and printer access
 - Full array of mathematical, string handling, and graphics functions
 - Can handle fixed and dynamic variable and control arrays
 - Sequential and random access file support
 - Useful debugger and error-handling facilities
 - Powerful database access tools
 - ActiveX support
 - Package & Deployment Wizard makes distributing your applications simple

Visual Basic 6.0 versus Other Versions of Visual Basic

- The original Visual Basic for DOS and Visual Basic For Windows were introduced in 1991.
- Visual Basic 3.0 (a vast improvement over previous versions) was released in 1993.

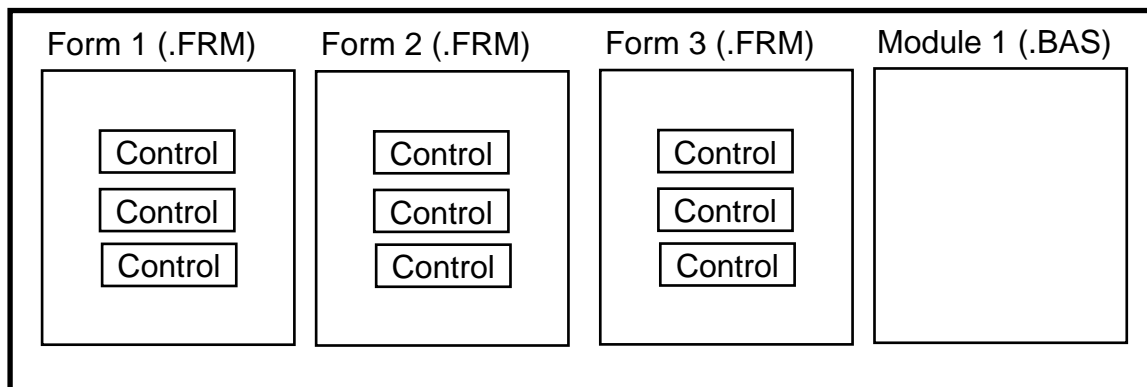
- Visual Basic 4.0 released in late 1995 (added 32 bit application support).
- Visual Basic 5.0 released in late 1996. New environment, supported creation of ActiveX controls, deleted 16 bit application support.
- Visual Basic 6.0 - some identified new features of Visual Basic 6.0:
 - Faster compiler
 - New ActiveX data control object
 - Allows database integration with wide variety of applications
 - New data report designer
 - New Package & Deployment Wizard
 - Additional internet capabilities

16 Bits versus 32 Bits

- Applications built using the Visual Basic 3.0 and the 16 bit version of Visual Basic 4.0 will run under Windows 3.1, Windows for Workgroups, Windows NT, or Windows 95
- Applications built using the 32 bit version of Visual Basic 4.0, Visual Basic 5.0 and Visual Basic 6.0 will only run with Windows 95 or Windows NT (Version 3.5.1 or higher).
- In this class, we will use Visual Basic 6.0 under Windows 95 OR higher version, recognizing such applications will not operate in 16 bit environments.

Structure of a Visual Basic Application

Project (.VBP, .MAK)



Application (Project) is made up of:

- **Forms** - Windows that you create for user interface
- **Controls** - Graphical features drawn on forms to allow user interaction (text boxes, labels, scroll bars, command buttons, etc.) (Forms and Controls are **objects**.)
- **Properties** - Every characteristic of a form or control is specified by a property. Example properties include names, captions, size, color, position, and contents. Visual Basic applies default properties. You can change properties at design time or run time.
- **Methods** - Built-in procedure that can be invoked to impart some action to a particular object.

- **Event Procedures** - Code related to some object. This is the code that is executed when a certain event occurs.
- **General Procedures** - Code not related to objects. This code must be invoked by the application.
- **Modules** - Collection of general procedures, variable declarations, and constant definitions used by application.

Steps in Developing Application

- There are three primary steps involved in building a Visual Basic application:
 1. **Draw the user interface**
 2. **Assign properties** to controls
 3. **Attach code** to controls

We'll look at each step.

Drawing the User Interface and Setting Properties

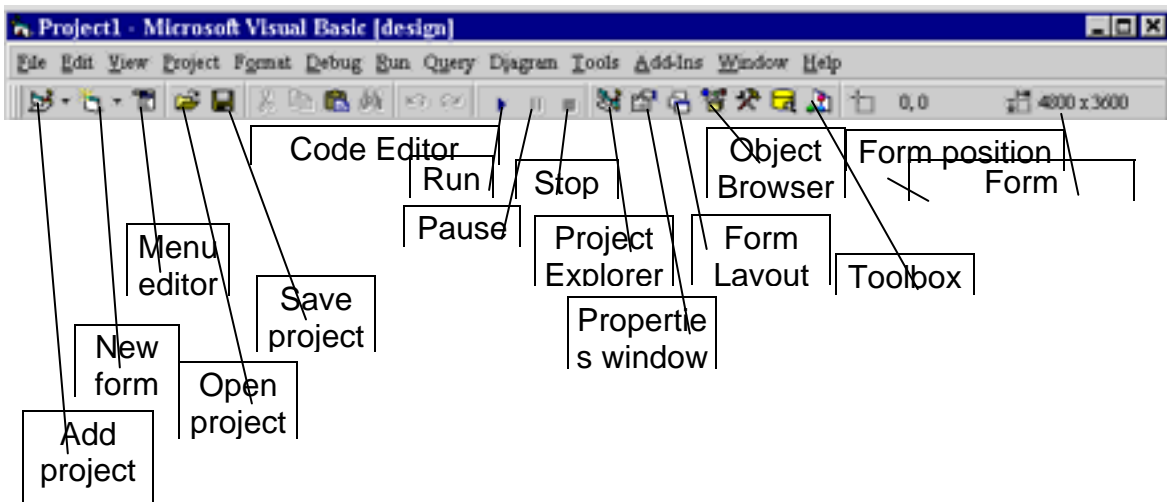
Visual Basic operates in three modes.

- **Design** mode - used to build application
- **Run** mode - used to run the application
- **Break** mode - application halted and debugger is available

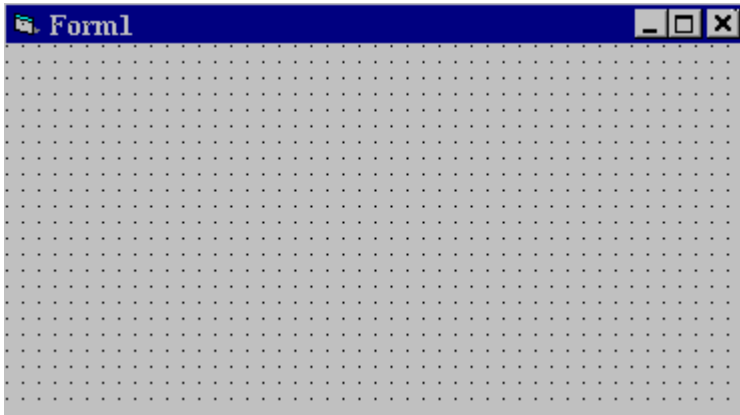
We focus here on the **design** mode.

Six windows appear when you start Visual Basic.

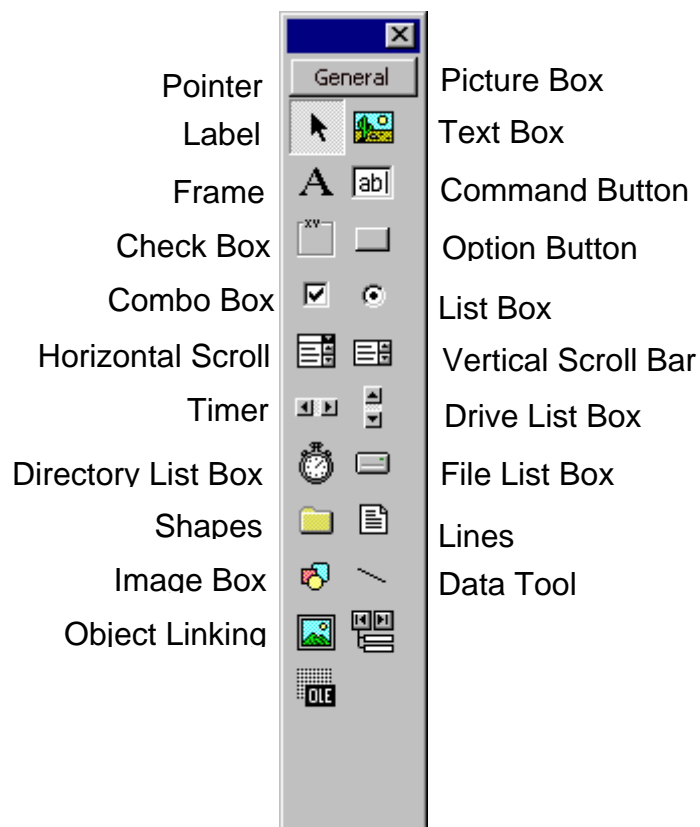
- The **Main Window** consists of the title bar, menu bar, and toolbar. The title bar indicates the project name, the current Visual Basic operating mode, and the current form. The menu bar has drop-down menus from which you control the operation of the Visual Basic environment. The toolbar has buttons that provide shortcuts to some of the menu options. The main window also shows the location of the current form relative to the upper left corner of the screen (measured in twips) and the width and length of the current form.



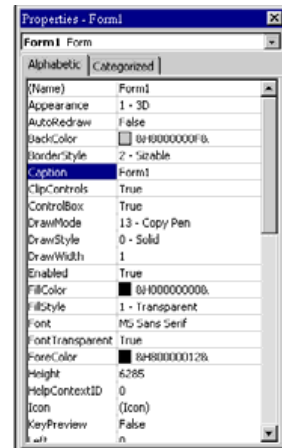
- The **Form Window** is central to developing Visual Basic applications. It is where you draw your application.



- The **Toolbox** is the selection menu for controls used in your application.



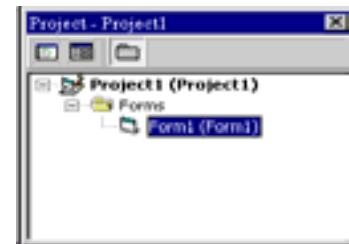
- The **Properties Window** is used to establish initial property values for objects. The drop-down box at the top of the window lists all objects in the current form. Two views are available: Alphabetic and Categorized. Under this box are the available properties for the currently selected object.



- The **Form Layout Window** shows where (upon program execution) your form will be displayed relative to your monitor's screen:



- The **Project Window** displays a list of all forms and modules making up your application. You can also obtain a view of the **Form** or **Code** windows (window containing the actual Basic coding) from the Project window.



- As mentioned, the user interface is 'drawn' in the form window. There are two ways to place controls on a form:
 1. Double-click the tool in the toolbox and it is created with a default size on the form. You can then move it or resize it.
 2. Click the tool in the toolbox, then move the mouse pointer to the form window. The cursor changes to a crosshair. Place the crosshair at the upper left corner of where you want the control to be, press the left mouse button and hold it down while dragging the cursor toward the lower right corner. When you release the mouse button, the control is drawn.
- To **move** a control you have drawn, click the object in the form window and drag it to the new location. Release the mouse button.
- To **resize** a control, click the object so that it is select and sizing handles appear. Use these handles to resize the object.

Click here to move

Use sizing handles to resize

Setting Properties of Objects at Design Time

Each form and control has **properties** assigned to it by default when you start up a new project. There are two ways to display the properties of an object. The first way is to click on the object (form or control) in the form window. Then, click on the Properties Window or the Properties Window button in the tool bar. The second way is to first click on the Properties Window. Then, select the object from the **Object** box in the Properties Window. Shown is the Properties Window for the stopwatch application:



The drop-down box at the top of the Properties Window is the **Object** box. It displays the name of each object in the application as well as its type. This display shows the **Form** object. The **Properties** list is directly below this box. In this list, you can scroll through the list of properties for the selected object. You may select a property by clicking on it. Properties can be changed by typing a new value or choosing from a list of predefined settings (available as a drop down list). Properties can be viewed in two ways: **Alphabetic** and **Categorized**.

A very important property for each object is its **name**. The name is used by Visual Basic to refer to a particular object in code.

A convention has been established for naming Visual Basic objects. This convention is to use a three letter prefix (depending on the object) followed by a name you assign. A few of the prefixes are (we'll see more as we progress in the class):

Form	frm	frmWatch
Command Button	cmd, btn	cmdExit, btnStart
Label	lbl	lblStart, lblEnd
Text Box	txt	txtTime, txtName
Menu	mnu	mnuExit, mnuSave
Check box	chk	chkChoice

Object	Prefix	Example
Form	frm	frmWatch
Command Butto	cmd, btn	cmdExit, btnStart
Label	lbl	lblStart, lblEnd
Text Box	txt	txtTime, txtName
Menu	mnu	mnuExit, mnuSave
Check box	chk	chkChoice

Object names can be up to 40 characters long, must start with a letter, must contain only letters, numbers, and the underscore (_) character. Names are used in setting properties at run time and also in establishing procedure names for object events.

Ex 1 Program for add two Number

1. First step -Draw the user interface

Put two label , three text box and one command button



2. Second Step-Assign properties to controls

label1

Name lblFirst
Caption Enter First Number

Label2

Name lblSecond
Caption Enter Second Number

Text1

Name txtFirst

Text

Text2

Name txtSecond

Text



Text3
Name txtAns
Text

Command1
Name cmdAdd
Caption Addition

Form
Name **FrmAddition**
Caption **Addition**

Third Step- Attach code to controls

1. Double click addition buttn
you will see following two lins

Private Sub CmdAdd_Click()

End Sub

Private- that means the action takes place only within procedure

Sub- mean sub routing or procedure

cmdAdd_Click()- mean CmdAdd buttion action event this event is click

End Sub- end the procedure

Attach code as follows

```
Private Sub CmdAdd_Click()  
    txtAns = Val(TxtFirst) + Val(TxtSecond)  
End Sub
```

Run Project press F5 or Click run button or Click Start on the run menu

Setting Properties at Run Time

You can also set or modify properties while your application is running. To do this, you must write some code. The code format is:

ObjectName.Property = NewValue

Such a format is referred to as dot notation. For example, to change the **BackColor** property of a form name **frmStart**, we'd type:

frmStart.BackColor = BLUE

How Names are used in Object Events

The names you assign to objects are used by Visual Basic to set up a framework of event-driven procedures for you to add code to. The format for each of these subroutines (all object procedures in Visual Basic are subroutines) is:

```
Sub ObjectName_Event (Optional Arguments)
    .
    .
End Sub
```

Visual Basic provides the **Sub** line with its arguments (if any) and the **End Sub** statement. You provide any needed code.

In the **Caption** properties of the three command buttons, notice the ampersand (&). The ampersand precedes a button's **access key**. That is, in addition to clicking on a button to invoke its event, you can also press its access key (no need for a mouse). The access key is pressed in conjunction with the **Alt** key. Hence, to invoke 'Begin Timing', you can either click the button or press Alt+B. Note in the button captions on the form, the access keys appear with an underscore (_).

Handling some of the common controls

The Text Box

The text box is the standard control that is used to receive input from the user as well as to display the output. It can handle string (text) and numeric data but not images or pictures. String in a text box can be converted to a numeric data by using the function Val(text). The following example illustrates a simple program that processes the inputs from the user.

Example

In this program, two text boxes are inserted into the form together with a few labels. The two text boxes are used to accept inputs from the user and one of the labels will be used to display the sum of two numbers that are entered into the two text boxes. Besides, a command button is also programmed to calculate the sum of the two numbers using the plus operator. The program use creates a variable sum to accept the summation of values from text box 1 and text box 2. The procedure to calculate and to display the output on the label is shown below. The output is shown in bellow

```
Private Sub Command1_Click()
```

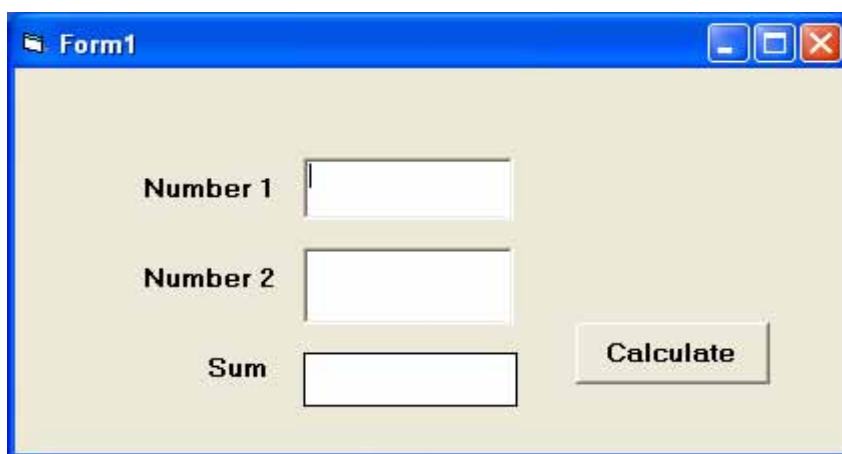
```
'To add the values in text box 1 and text box 2
```

```
Sum = Val(Text1.Text) + Val(Text2.Text)
```

```
'To display the answer on label 1
```

```
Label1.Caption = Sum
```

```
End Sub
```



3.2.2 The Label

The label is a very useful control for Visual Basic, as it is not only used to provide instructions and guides to the users, it can also be used to display outputs. One of its most important properties is Caption. Using the syntax label.Caption, it can display text and numeric data . You can change its caption in the properties window and also at runtime.

3.2.3 The Command Button

The command button is a very important control as it is used to execute commands. It displays an illusion that the button is pressed when the user click on it. The most common event associated with the command button is the Click event, and the syntax for the procedure is

```
Private Sub Command1_Click ()
```

```
    Statements
```

```
    Statements
```

```
    Statements
```

```
    Statements
```

```
End Sub
```

Visual Basic Statements and Expressions

The simplest statement is the **assignment** statement. It consists of a variable name, followed by the assignment operator (=), followed by some sort of **expression**.

Examples:

```
StName="Saman"  
Speed=60  
Energy = Mass * LIGHTSPEED ^ 2  
NetWorth = Assets - Liabilities
```

The assignment statement stores information.

- Statements normally take up a single line with no terminator. Statements can be **stacked** by using a colon (:) to separate them. Example:

```
StartTime = Now : EndTime = StartTime + 10
```

(Be careful stacking statements, especially with If/End If structures. You may not get the response you desire.)

- If a statement is very long, it may be continued to the next line using the **continuation** character, an underscore (_). Example:

Months = Log(Final * IntRate / Deposit + 1) _ / Log(1 + IntRate)

Comment

Comment statements begin with the keyword **Rem** or a single quote ('). For example:

```
Rem This is a remark  
' This is also a remark  
x = 2 * y ' another way to write a remark or comment
```

You, as a programmer, should decide how much to comment your code. Consider such factors as reuse, your audience, and the legacy of your code.

Variables

- We're now ready to attach code to our application. As objects are added to the form, Visual Basic automatically builds a framework of all event procedures. We simply add code to the event procedures we want our application to respond to. But before we do this, we need to discuss **variables**.
- Variables are used by Visual Basic to hold information needed by your application. Rules used in naming variables:

A variable name:

- **Must begin with a letter.**
- **Must not exceed 255 characters**
- **Can't contain an embedded period or embedded type-declaration character.**
- **Must be unique within the same scope, which is the range from which the variable can be referenced — a procedure, a form, and so on.**
- **They may include letters, numbers, and underscore (_)**
- **You cannot use a reserved word (word needed by Visual Basic)**

Variable Declaration

There are three ways for a variable to be typed (declared):

1. Default
2. Implicit
3. Explicit

Default

Dim variablename

Variables declared with the Dim statement within a procedure exist only as long as the procedure is executing. When the procedure finishes, the value of the variable disappears. In addition, the value of a variable in a procedure is local to that procedure — that is, you can't access a variable in one procedure from another procedure. These characteristics allow you to use the same variable names in different procedures without worrying about conflicts or accidental changes.

Static Variables declaration

Static <variablename >

Variables have a lifetime, the period of time during which they retain their value. The values in module-level and public variables are preserved for the lifetime of your application. However, local variables declared with Dim exist only while the procedure in which they are declared is executing. Usually, when a procedure is finished executing, the values of its local variables are not preserved and the memory used by the local variables is reclaimed. The next time the procedure is executed, all its local variables are reinitialized.

However, you can preserve the value of a local variable by making the variable static. Use the Static keyword to declare one or more variables inside a procedure, exactly as you would with the Dim statement:

Eg

Static Depth

If variables are not implicitly or explicitly typed, they are assigned the **variant** type by default. The variant data type is a special type used by Visual Basic **that can contain numeric, string, or date data.**

```
Private Sub Command1_Click()  
    Dim ProName  
    Dim Quantity, ProCode, Salesprice  
    ProName = "Book"  
    Quantity = 200  
    ProCode = "AI001"  
    Salesprice = 2000.75  
    Debug.Print ProName; Quantity, ProCode, Salesprice  
End Sub
```

```
Output  
Book 200   AI001   2000.75
```

Implicitly

To **implicitly** type a variable, use the corresponding suffix shown above in the data type table. For example,

creates a string variable

```
TextValue$ = "This is a string"
```

creates an integer variable.

```
Amount% = 300
```

Do not need declare variable before use it

```
Private Sub Command1_Click()  
    ProName = "Book"  
    Quantity = 20  
    ProCode = "AI001"  
    SalesPrice = 200.75  
    Tax = 50  
    Total = Quantity * SalesPrice + Tax  
    Debug.Print "Proname", "Quantity", "SalesPrice", "Total Price"  
    Debug.Print ProName, Quantity, SalesPrice, Total  
End Sub
```

Output

Proname	Quantity	SalesPrice	Total Price
Book	20	200.75	4065

Note

Visual Basic automatically creates a variable with that name, which you can use as if you had explicitly declared it. While this is convenient, it can lead to subtle errors in your code if you misspell a variable name.

```
Private Sub Command1_Click()  
    ProName = "Book"  
    Quantity = 20  
    ProCode = "AI001"  
    SalesPrice = 200.75  
    Tax = 50  
    Total = Quantity * SalesPrice + Txa  
    Debug.Print "Proname", "Quantity", "SalesPrice", "Total Price"  
    Debug.Print ProName, Quantity, SalesPrice, Total  
End Sub
```

eg.

Proname	Quantity	SalesPrice	Total Price
Book	20	200.75	4015

Explicitly

There are many advantages to **explicitly** typing variables. Primarily, we insure all computations are properly done, mistyped variable names are easily spotted, and Visual Basic will take care of insuring consistency in upper and lower case letters used in variable names. Because of these advantages, and because it is good programming practice, we will explicitly type all variables.

To explicitly declare variables

Type

Option Explicit - in the Declarations section of a class, form, or standard module:

–or–

From the **Tools menu**, choose **Options**, click the **Editor** tab and check the **Require Variable Declaration** option. This automatically inserts the Option Explicit statement in any new modules, but not in modules already created; therefore, you must manually add Option Explicit to any existing modules within a project.

To **explicitly** type a variable, you must first determine its **scope**. There are four levels of scope:

- Procedure level
- Procedure level, static
- Form and module level
- Global level

Within a procedure, variables are declared using the **Dim** statement:

```
Dim MyInt as Integer
Dim MyDouble as Double
Dim MyString, YourString as String
```

Procedure level variables declared in this manner do not retain their value once a procedure terminates.

To make a procedure level variable retain its value upon exiting the procedure, replace the Dim keyword with **Static**:

```
Static MyInt as Integer
Static MyDouble as Double
```

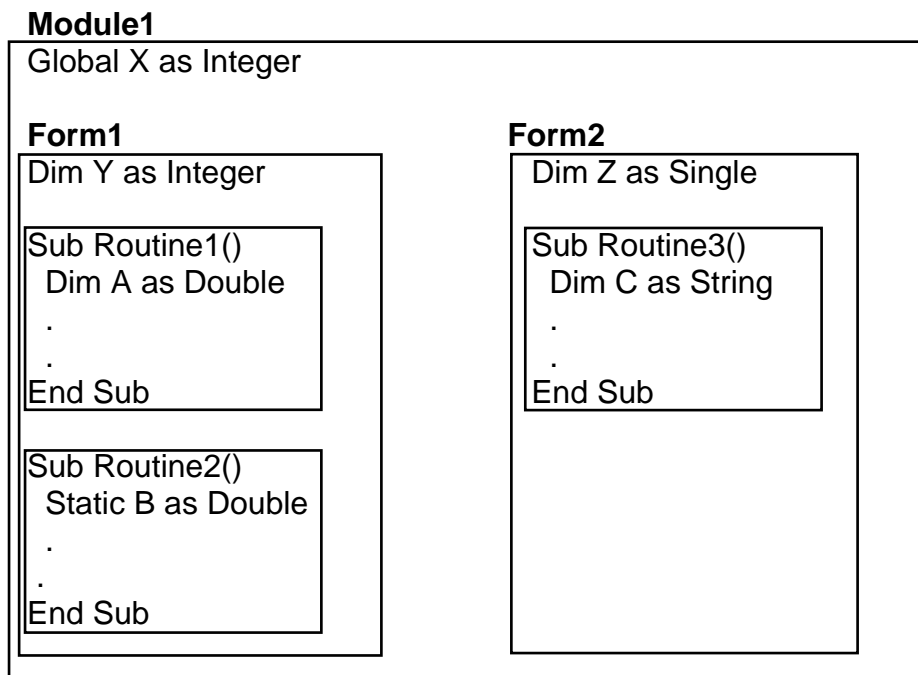
Form (module) level variables retain their value and are available to all procedures within that form (module). Form (module) level variables are declared in the **declarations** part of the **general** object in the form's (module's) code window. The **Dim** keyword is used:

```
Dim MyInt as Integer
Dim MyDate as Date
```

Global level variables retain their value and are available to all procedures within an application. Module level variables are declared in the **declarations** part of the **general** object of a module's code window. (It is advisable to keep all global variables in one module.) Use the **Global** keyword:

```
Global MyInt as Integer
Global MyDate as Date
```

- What happens if you declare a variable with the same name in two or more places? More local variables **shadow** (are accessed in preference to) less local variables. For example, if a variable MyInt is defined as Global in a module and declared local in a routine MyRoutine, while in MyRoutine, the local value of MyInt is accessed. Outside MyRoutine, the global value of MyInt is accessed.
- Example of Variable Scope:



Procedure Routine1 has access to X, Y, and A (loses value upon termination)
 Procedure Routine2 has access to X, Y, and B (retains value)
 Procedure Routine3 has access to X, Z, and C (loses value)

• Data types

Integer	}	Integer
Long		
Single	}	Floating point
double		
Currency		-Number with fixed decimal point of four position
Byte		-Store 0 to 255 one character
Date		-Store dates and times
String		-Store textual data
Boolean		-Store boolean value true false
Variant		-Can Store any of the above

**Constant value
 is a fixed value
 Declaration
 const pi=3.141593**

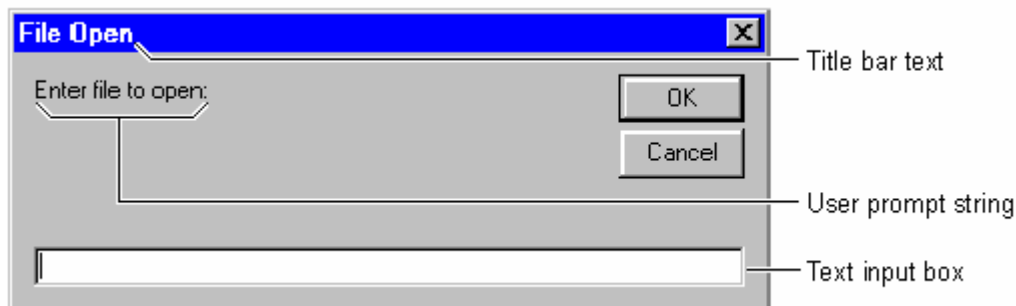
Data Type	Suffix
Boolean	None
Integer	%
Long (Integer)	&
Single (Floating)	!
Double (Floating)	#
Currency	@
Date	None
Object	None
String	\$
Variant	None

Use this function	To do this
InputBox function	Display a command prompt in a dialog box, and return whatever is entered by the user.
MsgBox function	Display a message in a dialog box, and return a value indicating the command button was clicked by the user.

Predefined Dialog Boxes

Syntex

1. Inputbox(Prompt String,Diolog Title)



The easiest way to add a dialog box to your application is to use a predefined dialog, because you don't have to worry about designing, loading, or showing the dialog box. However, your control over its appearance is limited. Predefined dialog boxes are always modal.

The following table lists the functions you can use to add predefined dialog boxes to your Visual Basic application.

Prompting for Input with InputBox

Use the InputBox function to solicit data from the user. This function displays a modal dialog box that asks the user to enter some data. The text input box prompts the user for the name of the file to open.

The following code displays the input box

```
FileName = InputBox("Enter file to open:", "File Open")
```

Note Remember that when you use the InputBox function, you have little control over the components of the dialog box. You can change only the text in the title bar, the command prompt displayed to the user, the position of the dialog box on the screen, and whether or not it displays a Help button.

Box

2. The MessageBox

One of the best functions in Visual Basic is the message box. The message box displays a message, optional icon, and selected set of command buttons. The user responds by clicking a button.

The statement form of the message box returns no value (it simply displays the box):

```
MsgBox (Message, Type, Title)
```

where

Message	Text message to be displayed
Type	Type of message box (discussed in a bit)
Title	Text in title bar of message box

You have no control over where the message box appears on the screen.

The function form of the message box returns an integer value (corresponding to the button clicked by the user). Example of use (Response is returned value):

```
Dim Response as Integer  
Response = MsgBox(Message, Type, Title)
```

- The Type argument is formed by summing four values corresponding to the buttons to display, any icon to show, which button is the default response, and the modality of the message box.

- The first component of the Type value specifies the buttons to display:

Value	Meaning	Symbolic Constant
0	OK button only	vbOKOnly
1	OK/Cancel buttons	vbOKCancel
2	Abort/Retry/Ignore buttons	vbAbortRetryIgnore
3	Yes/No/Cancel buttons	vbYesNoCancel
4	Yes/No buttons	vbYesNo
5	Retry/Cancel buttons	vbRetryCancel

- The second component of Type specifies the icon to display in the message box:

Value	Meaning	Symbolic Constant
0	No icon	(None)
16	Critical icon	vbCritical
32	Question mark	vbQuestion
48	Exclamation point	vbExclamation
64	Information icon	vbInformation

The third component of Type specifies which button is default (i.e. pressing Enter is the same as clicking the default button):

Value	Meaning	Symbolic Constant
0	First button default	vbDefaultButton1
256	Second button default	vbDefaultButton2
512	Third button default	vbDefaultButton3

The fourth and final component of Type specifies the modality:

Value	Meaning	Symbolic Constant
0	Application modal	vbApplicationModal
4096	System modal	vbSystemModal

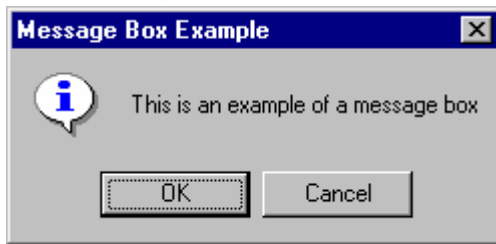
If the box is Application Modal, the user must respond to the box before continuing work in the current application. If the box is System Modal, all applications are suspended until the user responds to the message box.

Note for each option in Type, there are numeric values listed and symbolic constants. Recall, it is strongly suggested that the symbolic constants be used instead of the numeric values. You should agree that vbOKOnly means more than the number 0 when selecting the button type.

The value returned by the function form of the message box is related to the button clicked:

Value	Meaning	Symbolic Constant
1	OK button selected	vbOK
2	Cancel button selected	vbCancel
3	Abort button selected	vbAbort
4	Retry button selected	vbRetry
5	Ignore button selected	vbIgnore
6	Yes button selected	vbYes
7	No button selected	vbNo

- Message Box Example:



MsgBox("This is an example of a message box", vbOKCancel + vbInformation, "Message Box Example")

MsgBox("This is Message ", 4 + 32, "hjkhjk")

- **Visual Basic Operators**

1. **Arithmetic. Operators**
2. **String -Concatentate Operators**
3. **Comparison Operators**
4. **Logical Operators**

Arithmetic. Operators

Operator	Operation
^	Exponentiation
* /	Multiplication and division
\	Integer division (truncates)
Mod	Modulus
+ -	Addition and subutraction

Parentheses around expressions can change precedence.

String -Concatentate Operators

To **concatentate** two strings, use the **&** symbol or the **+** symbol:

```
lblTime.Caption = "The current time is" & Format(Now, "hh:mm")
txtSample.Text = "Hook this " + "to this"
```

- There are six **comparison** operators in Visual Basic:

Operator	Comparison
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
=	Equal to
<>	Not equal to

The result of a comparison operation is a Boolean value (**True** or **False**).

Three **logical** operators

Operator	Operation
Not	Logical not
And	Logical and
Or	Logical or

The **Not** operator simply negates an operand.

The **And** operator returns a True if both operands are True. Else, it returns a False.

The **Or** operator returns a True if either of its operands is True, else it returns a False.

- Logical operators follow arithmetic operators in precedence.

Example 2

Savings Account

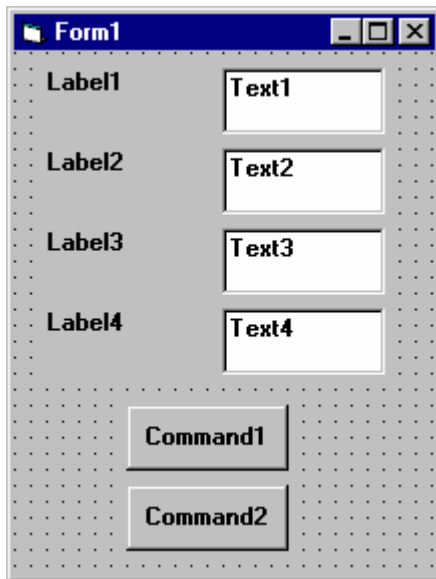
1. Start a new project. The idea of this project is to determine how much you save by making monthly deposits into a savings account. For those interested, the mathematical formula used is:

$$F = D [(1 + I)^M - 1] / I$$

where

F - Final amount
D - Monthly deposit amount
I - Monthly interest rate
M - Number of months

- Place 4 label boxes, 4 text boxes, and 2 command buttons on the form. It should look something like this:



- Set the properties of the form and each object.

Form1:

BorderStyle	1-Fixed Single
Caption	Savings Account
Name	frmSavings

Label1:

Caption	Monthly Deposit
---------	-----------------

Label2:

Caption	Yearly Interest
---------	-----------------

Label3:

Caption	Number of Months
---------	------------------

Label4:

Caption	Final Balance
---------	---------------

Text1:

Text	[Blank]
Name	txtDeposit

Text2:

Text	[Blank]
Name	txtInterest

Text3:

Text	[Blank]
Name	txtMonths

Text4:

Text	[Blank]
Name	txtFinal

Command1:

Caption	&Calculate
Name	cmdCalculate

Command2:

Caption	E&xit
Name	cmdExit

Now, your form should look like this:

4. Declare four variables in the **general declarations** area of your form. This makes them available to all the form procedures:

Option Explicit**Dim Deposit As Single****Dim Interest As Single****Dim Months As Single****Dim Final As Single**

The **Option Explicit** statement forces us to declare all variables.

5. Attach code to the **cmdCalculate** command button **Click** event.

```

Private Sub cmdCalculate_Click ()
    Dim IntRate As Single
    'Read values from text boxes
    Deposit = Val(txtDeposit.Text)
    Interest = Val(txtInterest.Text)
    IntRate = Interest / 12.00
    Months = Val(txtMonths.Text)
    'Compute final value and put in text box
    Final = Deposit * ((1 + IntRate) ^ Months - 1) / IntRate
    txtFinal.Text = Format(Final, "#####0.00")
End Sub

```

This code reads the three input values (monthly deposit, interest rate, number of months) from the text boxes, computes the final balance using the provided formula, and puts that result in a text box.

6. Attach code to the **cmdExit** command button **Click** event.

```
Private Sub cmdExit_Click ()  
    End  
End Sub
```

7. Play with the program. Make sure it works properly. Save the project.

Data Types more

Date variables are stored as IEEE 64-bit (8-byte) floating-point numbers that represent dates ranging from 1 January 100 to 31 December 9999 and times from 0:00:00 to 23:59:59. Any recognizable literal date values can be assigned to Date variables. Date literals must be enclosed within number signs (#), for example, #January 1, 1993# or #1 Jan 93#.

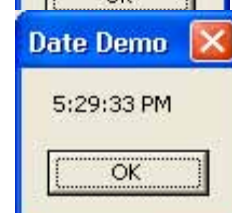
Date variables display dates according to the short date format recognized by your computer. Times display according to the time format (either 12-hour or 24-hour) recognized by your computer.

When other numeric types are converted to Date, values to the left of the decimal represent date information while values to the right of the decimal represent time. Midnight is 0 and midday is 0.5. Negative whole numbers represent dates before 30 December 1899.

```
Dim thedate As Date  
thedata = Date  
Res = MsgBox(Date, vbOKOnly, "Date Demo")
```



```
Dim thedate As Date  
thedata = Time  
Res = MsgBox(Time, vbOKOnly, "Date Demo")
```



```
Dim thedate As Date  
thedata = Date & Time  
Res = MsgBox(thedata, vbOKOnly, "Date Demo")
```



```
Dim thedate1, thedate2 As Date  
thedata1 = Day(Date)  
  
Res = MsgBox(thedata1, vbOKOnly, "Date Demo")
```



Syntax

Weekday(date, [firstdayofweek])

The Weekday function syntax has these named arguments:

Part	Description
date	Required. Variant, numeric expression, string expression, or any combination, that can represent a date. If date contains Null, Null is returned.
firstdayofweek	Optional. A constant that specifies the first day of the week. If not specified, vbSunday is assumed.

Settings

The firstdayofweek argument has these settings:

Return Values

The Weekday function can return any of these values:

Constant	Value	Description
vbSunday	1	Sunday
vbMonday	2	Monday
vbTuesday	3	Tuesday
vbWednesday	4	Wednesday
vbThursday	5	Thursday
vbFriday	6	Friday
vbSaturday	7	Saturday

Year(date)

Formatting Numbers, Dates, and Times

Visual Basic provides great flexibility in displaying number formats, as well as date and time formats. You can easily display international formats for numbers, dates, and times.

The Format function converts the numeric value to a text string and gives you control over the string's appearance. For example, you can specify the number of decimal places, leading or trailing zeros, and currency formats. The syntax is:

Format(expression[, format[, firstdayofweek[, firstweekofyear]])

The expression argument specifies a number to convert, and the format argument is a string made up of symbols that shows how to format the number. The most commonly used symbols are listed in the table below.

Symbol	Description
0	Digit placeholder; prints a trailing or a leading zero in this position, if appropriate.
#	Digit placeholder; never prints trailing or leading zeros.
.	Decimal placeholder.
,	Thousands separator.
– + \$ () space	Literal character; characters are displayed exactly as typed into the format string.

The *firstdayofweek* argument is a constant that specifies the first day of the week; the *firstweekofyear* argument is a constant that specifies the first week of the year. Both arguments are optional. For more information about these constants, see

Named **Formats**

Visual Basic provides several standard formats to use with the Format function. Instead of designating symbols in the format argument, you specify these formats by name in the format argument of the Format function. Always enclose the format name in double quotation marks ("").

The following table lists the format names you can use.

Named format	Description
General Number	Displays number with no thousand separator.
Currency	Displays number with thousand separator, if appropriate; display two digits to the right of the decimal separator. Output is based on user's system settings.
Fixed	Displays at least one digit to the left and two digits to the right of the decimal separator.
Standard	Displays number with thousand separator, at least one digit to the left and two digits to the right of the decimal separator.
Percent	Multiplies the value by 100 with a percent sign at the end.
Scientific	Uses standard scientific notation.
General Date	Shows date and time if expression contains both. If expression is only a date or a time, the missing information is not displayed. Date display is determined by user's system settings.
Long Date	Uses the Long Date format specified by user's system settings.

Medium Date	Uses the dd-mmm-yy format (for example, 03-Apr-93). Date display is determined by user's system settings.
Short Date	Uses the Short Date format specified by user's system settings.
Long Time	Displays a time using user's system's long-time format; includes hours, minutes, seconds.
Medium Time	Shows the hour, minute, and "AM" or "PM" using the "hh:mm AM/PM" format.
Short Time	Shows the hour and minute using the hh:mm format.
Yes/No	Any nonzero numeric value (usually –1) is Yes. Zero is No.
True/False	Any nonzero numeric value (usually –1) is True. Zero is False.
On/Off	Any nonzero numeric value (usually –1) is On. Zero is Off.

The Format function supports many other special characters, such as the percentage placeholder and exponents.

Number Formats

The following number conversions assume that the country in the Windows Control Panel is set to "English (United States)."

Format syntax	Result
Format(8315.4, "00000.00")	08315.40
Format(8315.4, "#####.##")	8315.4
Format(8315.4, "##,##0.00")	8,315.40
Format(315.4, "\$##0.00")	\$315.40

The symbol for the decimal separator is a period (.), and the symbol for the thousands separator is a comma (,). However, the separator character that is actually displayed depends on the country specified in the Windows Control Panel.

Printing Formatted Dates and Times

To print formatted dates and times, use the Format function with symbols representing date and time. These examples use the Now and Format functions to identify and format the current date and time. The following examples assume that the Regional Settings dialog box of the Windows Control Panel is set to "English(United States)".

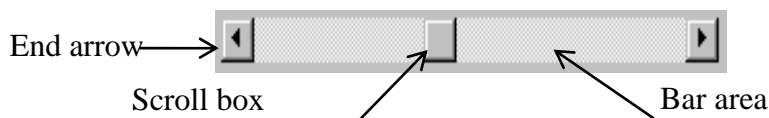
Format syntax	Result
Format(Now, "m/d/yy")	1/27/93
Format(Now, "dddd, mmmm dd, yyyy")	Wednesday, January 27, 1993
Format(Now, "d-mmm")	27-Jan
Format(Now, "mmmm-yy")	January-93
Format(Now, "hh:mm AM/PM")	07:18 AM
Format(Now, "h:mm:ss a/p")	7:18:00 a
Format(Now, "d-mmmm h:mm")	3-January 7:18

Horizontal and Vertical Scroll Bars



Horizontal and vertical scroll bars are widely used in Windows applications. Scroll bars provide an intuitive way to move through a list of information and make great input devices.

Both type of scroll bars are comprised of three areas that can be clicked, or dragged, to change the scroll bar value. Those areas are:



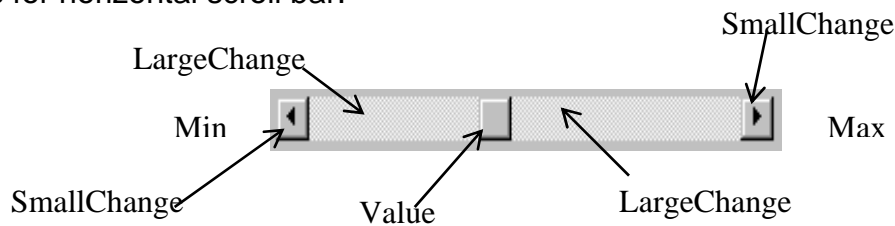
Clicking an end arrow increments the scroll box a small amount, clicking the bar area increments the scroll box a large amount, and dragging the scroll box (thumb) provides continuous motion. Using the properties of scroll bars, we can completely specify how one works. The scroll box position is the only output information from a scroll bar.

Scroll Bar Properties:

- LargeChange** Increment added to or subtracted from the scroll bar Value property when the bar area is clicked.
- Max** The value of the horizontal scroll bar at the far right and the value of the vertical scroll bar at the bottom. Can range from -32,768 to 32,767.
- Min** The other extreme value - the horizontal scroll bar at the left and the vertical scroll bar at the top. Can range from -32,768 to 32,767.
- SmallChange** The increment added to or subtracted from the scroll bar Value property when either of the scroll arrows is clicked.

Value The current position of the scroll box (thumb) within the scroll bar. If you set this in code, Visual Basic moves the scroll box to the proper position.

Properties for horizontal scroll bar:



Properties for vertical scroll bar:

A couple of important notes about scroll bars:

Note that although the extreme values are called Min and Max, they do not necessarily represent minimum and maximum values. There is nothing to keep the Min value from being greater than the Max value. In fact, with vertical scroll bars, this is the usual case. Visual Basic automatically adjusts the sign on the SmallChange and LargeChange properties to insure proper movement of the scroll box from one extreme to the other.

If you ever change the Value, Min, or Max properties in code, make sure Value is at all times between Min and Max or and the program will stop with an error message.

Scroll Bar Events:

- Change** Event is triggered after the scroll box's position has been modified. Use this event to retrieve the Value property after any changes in the scroll bar.
- Scroll** Event triggered continuously whenever the scroll box is being moved.

Check Boxes



- Check boxes provide a way to make choices from a list of potential candidates. Some, all, or none of the choices in a group may be selected.
- Check Box Properties:

Caption	Identifying text next to box.
Font	Sets font type, style, size.
Value	Indicates if unchecked (0, vbUnchecked), checked (1, vbChecked), or grayed out (2, vbGrayed).

- Check Box Events:

Click	Triggered when a box is clicked. Value property is automatically changed by Visual Basic.
--------------	---

Option Buttons



- Option buttons provide the capability to make a mutually exclusive choice among a group of potential candidate choices. Hence, option buttons work as a group, only one of which can have a True (or selected) value.
- Option Button Properties:

Caption	Identifying text next to button.
Font	Sets font type, style, size.
Value	Indicates if selected (True) or not (False). Only one option button in a group can be True. One button in each group of option buttons should always be initialized to True at design time.

- Option Button Events:

Click	Triggered when a button is clicked. Value property is automatically changed by Visual Basic.
--------------	--

Frames



- We've seen that both option buttons and check boxes work as a group. Frames provide a way of grouping related controls on a form. And, in the case of option buttons, frames affect how such buttons operate.
- To group controls in a frame, you first draw the frame. Then, the associated controls must be drawn in the frame. This allows you to move the frame and controls together. And, once a control is drawn within a frame, it can be copied and pasted to create a control array within that frame. To do this, first click on the object you want to copy. Copy the object. Then, click on the frame. Paste the object. You will be asked if you want to create a control array. Answer Yes.
- Drawing the controls outside the frame and dragging them in, copying them into a frame, or drawing the frame around existing controls will not result in a proper grouping. It is perfectly acceptable to draw frames within other frames.
- As mentioned, frames affect how option buttons work. Option buttons within a frame work as a group, independently of option buttons in other frames. Option buttons on the form, and not in frames, work as another independent group. That is, the form is itself a frame by default. We'll see this in the next example.
- It is important to note that an independent group of option buttons is defined by physical location within frames, not according to naming convention. That is, a control array of option buttons does not work as an independent group just because it is a control array. It would only work as a group if it were the only group of option buttons within a frame or on the form. So, remember physical location, and physical location only, dictates independent operation of option button groups.
- Frame Properties:

Caption	Title information at top of frame.
Font	Sets font type, style, size.

Control structures

Control structures allow you to control the flow of your program's execution. If left unchecked by control-flow statements, a program's logic will flow through statements from left to right, and top to bottom. While some very simple programs can be written with only this unidirectional flow, and while some flow can be controlled by using operators to regulate precedence of operations, most of the power and utility of any programming language comes from its ability to change statement order with structures and loops.

1. Branching
 - 1.1. IF
 - 1.2. Case
 - 1.3. Goto
2. Looping
 - 2.1. For next
 - 2.2. Do while
 - 2.3. Do until

1.1 IF Statement

Branching statements are used to cause certain actions within a program if a certain condition is met.

- Syntax

If <condition > Then statement

If Balance - Check < 0 Then Print "You are overdrawn"

Here, if and only if Balance - Check is less than zero, the statement "You are overdrawn" is printed.

If Then End If blocks to allow multiple statements:

Syntax

```
IF <condition> then
    statement1
    statement2
    .....
    statementn
end if
```

```
eg1:- Private Sub Command1_Click()
        number = Text1.Text
        If number Mod 2 = 0 Then
            MsgBox ("Even")
        End If
    End Sub
```

```

If Balance - Check < 0 Then
  Print "You are overdrawn"
  Print "Authorities have been notified"
End If

```

In this case, if Balance - Check is less than zero, two lines of information are printed.

Nested IF

eg2

```

IF Boolean Expression Then
  Statement1
  Statement2
  Statement2
  .....
  .....
  Statementn
Else
  IF Boolean Expression Then
    Statement1
    Statement2
    Statement2
    .....
    .....
    Statementn
  Else
    Statement1
    Statement2
    Statement2
    .....
    .....
    Statementn
  End If
End if

```

```

Private Sub Command1_Click()
  number = Text1.Text
  If number Mod 2 = 0 Then
    MsgBox ("Even")
  Else
    MsgBox ("odd")
  End If
End Sub

```

```

If Balance - Check < 0 Then
    Print "You are overdrawn"
    Print "Authorities have been notified"
Else
    Balance = Balance - Check
End If

```

Here, the same two lines are printed if you are overdrawn ($\text{Balance} - \text{Check} < 0$), but, if you are not overdrawn (**Else**), your new Balance is computed.

Or, we can add the **Elseif** statement:

```

If Balance - Check < 0 Then
    Print "You are overdrawn"
    Print "Authorities have been notified"
Elseif Balance - Check = 0 Then
    Print "Whew! You barely made it"
    Balance = 0
Else
    Balance = Balance - Check
End If

```

Now, one more condition is added. If your Balance equals the Check amount (**Elseif** $\text{Balance} - \text{Check} = 0$), a different message appears.

- In using branching statements, make sure you consider all viable possibilities in the If/Else/End If structure. Also, be aware that each If and Elseif in a block is tested sequentially. The first time an If test is met, the code associated with that condition is executed and the If block is exited. If a later condition is also True, it will never be considered.

Example

Savings Account -

1. Note the acceptable ASCII codes are 48 through 57 (numbers), 46 (the decimal point), and 8 (the backspace key). In the code, we use symbolic constants for the numbers and backspace key. Such a constant does not exist for the decimal point, so we will define one with the following line in the **general declarations** area:

```
Const vbKeyDecPt = 46
```

2. Add the following code to the three procedures: `txtDeposit_KeyPress`, `txtInterest_KeyPress`, and `txtMonths_KeyPress`.

```
Private Sub txtDeposit_KeyPress (KeyAscii As Integer)
'Only allow number keys, decimal point, or backspace
If (KeyAscii >= vbKey0 And KeyAscii <= vbKey9) Or KeyAscii = vbKeyDecPt Or
KeyAscii = vbKeyBack Then
    Exit Sub
Else
    KeyAscii = 0
    Beep
End If
End Sub
```

```
Private Sub txtInterest_KeyPress (KeyAscii As Integer)
'Only allow number keys, decimal point, or backspace
If (KeyAscii >= vbKey0 And KeyAscii <= vbKey9) Or KeyAscii = vbKeyDecPt Or
KeyAscii = vbKeyBack Then
    Exit Sub
Else
    KeyAscii = 0
    Beep
End If
End Sub
```

```
Private Sub txtMonths_KeyPress (KeyAscii As Integer)
'Only allow number keys, decimal point, or backspace
If (KeyAscii >= vbKey0 And KeyAscii <= vbKey9) Or KeyAscii = vbKeyDecPt Or
KeyAscii = vbKeyBack Then
    Exit Sub
Else
    KeyAscii = 0
    Beep
End If
End Sub
```

(In the If statements above, note the word processor causes a line break where there really shouldn't be one. That is, there is no line break between the words **Or KeyAscii** and **= vbKeyDecPt**. One appears due to page margins. In all code in these notes, always look for such things.)

3. Rerun the application and test the key trapping performance.

Select Case - Another Way to Branch

- In addition to If/Then/Else type statements, the **Select Case** format can be used when there are multiple selection possibilities.

```

Select Case marks
  Case 0 to 39
    grade="D"
  Case 39 to 40
    Grade="C"
  Case 41 to 50
    grade="C"
  Case 51 to 60
    Grade="B"
  Case Else
    Grade="A"
End Select

```

- Say we've written this code using the **If** statement:

```

If Age = 5 Then
  Category = "Five Year Old"
Elseif Age >= 13 and Age <= 19 Then
  Category = "Teenager"
Elseif (Age >= 20 and Age <= 35) Or Age = 50 Or (Age >= 60 and Age <= 65) Then
  Category = "Special Adult"
Elseif Age > 65 Then
  Category = "Senior Citizen"
Else
  Category = "Everyone Else"
End If

```

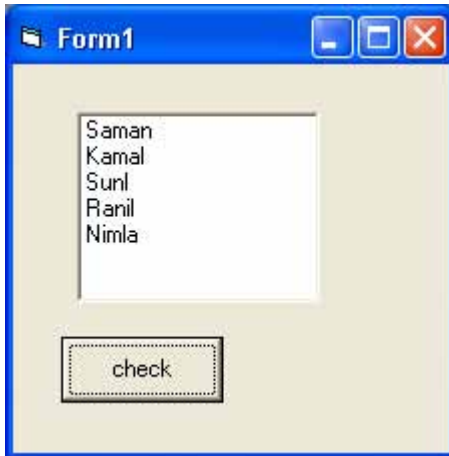
The corresponding code with **Select Case** would be:

```

Select Case Age
  Case 5
    Category = "Five Year Old"
  Case 13 To 19
    Category = "Teenager"
  Case 20 To 35, 50, 60 To 65
    Category = "Special Adult"
  Case Is > 65
    Category = "Senior Citizen"
  Case Else
    Category = "Everyone Else"
End Select

```

Notice there are several formats for the Case statement. Consult on-line help for discussions of these formats.

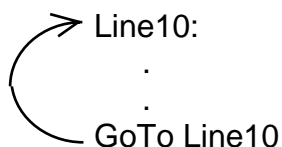


```
Private Sub Command1_Click()
    Select Case List1.ListIndex
        Case 0
            MsgBox ("Fiest name")
        Case 1
            MsgBox ("second name")
        Case 2
            MsgBox ("Thired name")
        Case 3
            MsgBox ("fourth name")
    End Select
End Sub
```

The GoTo Statement

Another branching statement, and perhaps the most hated statement in programming, is the **GoTo** statement. However, we will need this to do Run-Time error trapping. The format is **GoTo Label**, where **Label** is a labeled line. Labeled lines are formed by typing the *Label* followed by a colon.

- **GoTo** Example:



When the code reaches the GoTo statement, program control transfers to the line labeled Line10.

Visual Basic Looping

Looping is done with the **Do/Loop** format. Loops used for operations are to be repeated some number of times. The loop repeats until some specified condition at the beginning or end of the loop is met.

Do While/Loop Example:

```
Counter = 1
Do While Counter < 10
    Debug.Print Counter
    Counter = Counter + 1
Loop
```

This loop repeats as long as (**While**) the variable Counter is less than or equal to 1000. Note a Do While/Loop structure will not execute even once if the While condition is violated (False) the first time through. Also note the **Debug.Print** statement. What this does is print the value Counter in the Visual Basic Debug window. We'll learn more about this window later in the course.

Do Until/Loop Example:

```
Counter = 1
Do Until Counter > 1000
    Debug.Print Counter
    Counter = Counter + 1
Loop
```

This loop repeats **Until** the Counter variable exceeds 1000. Note a Do Until/Loop structure will not be entered if the Until condition is already True on the first encounter.

Do/Loop While Example:

```
Sum = 1
Do
    Debug.Print Sum
    Sum = Sum + 3
Loop While Sum <= 50
```

This loop repeats **While** the Variable Sum is less than or equal to 50. Note, since the While check is at the end of the loop, a Do/Loop While structure is always executed at least once.

Do/Loop Until Example:

```
Sum = 1
Do
    Debug.Print Sum
    Sum = Sum + 3
Loop Until Sum > 50
```

This loop repeats Until Sum is greater than 50. And, like the previous example, a Do/Loop Until structure always executes at least once.

Make sure you can always get out of a loop! Infinite loops are never nice. If you get into one, try **Ctrl+Break**. That sometimes works - other times the only way out is rebooting your machine!

The statement **Exit Do** will get you out of a loop and transfer program control to the statement following the Loop statement.

Visual Basic Counting

Counting is accomplished using the **For/Next** loop.

Example

```
For I = 1 to 50 Step 2
  A = I * 2
  Debug.Print A
Next I
```

In this example, the variable *I* initializes at 1 and, with each iteration of the For/Next loop, is incremented by 2 (**Step**). This looping continues until *I* becomes greater than or equal to its final value (50). If Step is not included, the default value is 1. Negative values of Step are allowed.

You may exit a For/Next loop using an **Exit For** statement. This will transfer program control to the statement following the **Next** statement.

Drive List Box



- The drive list box control allows a user to select a valid disk drive at run-time. It displays the available drives in a drop-down combo box. No code is needed to load a drive list box; Visual Basic does this for us. We use the box to get the current drive identification.

- Drive List Box Properties:

Drive	Contains the name of the currently selected drive.
-------	--

- Drive List Box Events:

Change	Triggered whenever the user or program changes the drive selection.
--------	---

Directory List Box



- The directory list box displays an ordered, hierarchical list of the user's disk directories and subdirectories. The directory structure is displayed in a list box. Like, the drive list box, little coding is needed to use the directory list box - Visual Basic does most of the work for us.

- Directory List Box Properties:

Path Contains the current directory path.

- Directory List Box Events:

Change Triggered when the directory selection is changed.

File List Box



- The file list box locates and lists files in the directory specified by its Path property at run-time. You may select the types of files you want to display in the file list box.

- File List Box Properties:

FileName Contains the currently selected file name.

Path Contains the current path directory.

Pattern Contains a string that determines which files will be displayed. It supports the use of * and ? wildcard characters. For example, using *.dat only displays files with the .dat extension.

- File List Box Events:

DbClick Triggered whenever a file name is double-clicked.

PathChange Triggered whenever the path changes in a file list box.

- You can also use the MultiSelect property of the file list box to allow multiple file selection.

Synchronizing the Drive, Directory, and File List Boxes

- The drive, directory, and file list boxes are almost always used together to obtain a file name. As such, it is important that their operation be synchronized to insure the displayed information is always consistent.

- When the drive selection is changed (drive box Change event), you should update the directory path. For example, if the drive box is named drvExample and the directory box is dirExample, use the code:

```
dirExample.Path = drvExample.Drive
```

- When the directory selection is changed (directory box Change event), you should update the displayed file names. With a file box named filExample, this code is:

```
filExample.Path = dirExample.Path
```

- Once all of the selections have been made and you want the file name, you need to form a text string that correctly and completely specifies the file identifier. This string concatenates the drive, directory, and file name information. This should be an easy task, except for one problem. The problem involves the backslash (\) character. If you are at the root directory of your drive, the path name ends with a backslash. If you are not at the root directory, there is no backslash at the end of the path name and you have to add one before tacking on the file name.
- Example code for concatenating the available information into a proper file name and then loading it into an image box is:

```
Dim YourFile as String
```

```
If Right(filExample.Path,1) = "\" Then
  YourFile = filExample.Path + filExample.FileName
Else
  YourFile = filExample.Path + "\" + filExample.FileName
End If
imgExample.Picture = LoadPicture(YourFile)
```

Note we only use properties of the file list box. The drive and directory box properties are only used to create changes in the file list box via code.

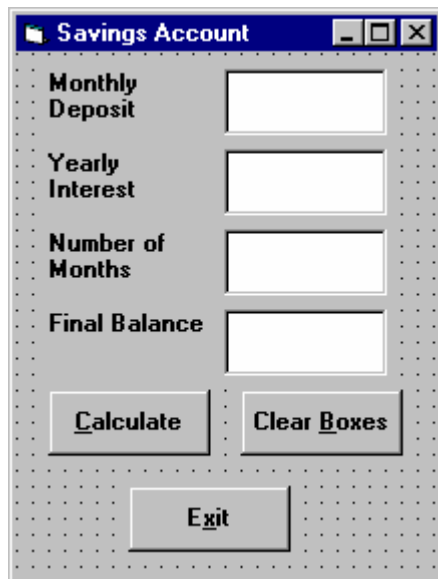
Example

Savings Account - Decisions

1. Here, we modify the Savings Account project to allow entering any three values and computing the fourth. First, add a third command button that will clear all of the text boxes. Assign the following properties:

Command3:	
Caption	Clear &Boxes
Name	cmdClear

The form should look something like this when you're done:



2. Code the **cmdClear** button **Click** event:

```

Private Sub cmdClear_Click ()
'Blank out the text boxes
txtDeposit.Text = ""
txtInterest.Text = ""
txtMonths.Text = ""
txtFinal.Text = ""
End Sub

```

This code simply blanks out the four text boxes when the **Clear** button is clicked.

3. Code the **KeyPress** event for the **txtFinal** object:

```

Private Sub txtFinal_KeyPress (KeyAscii As Integer)
'Only allow number keys, decimal point, or backspace
If (KeyAscii >= vbKey0 And KeyAscii <= vbKey9) Or KeyAscii = vbKeyDecPt Or
KeyAscii = vbKeyBack Then
Exit Sub
Else
KeyAscii = 0
Beep
End If
End Sub

```

We need this code because we can now enter information into the Final Value text box.

4. The modified code for the **Click** event of the **cmdCalculate** button is:

```
Private Sub cmdCalculate_Click()
Dim IntRate As Single
Dim IntNew As Single
Dim Fcn As Single, FcnD As Single
'Read the four text boxes
Deposit = Val(txtDeposit.Text)
Interest = Val(txtInterest.Text)
IntRate = Interest / 1200
Months = Val(txtMonths.Text)
Final = Val(txtFinal.Text)
'Determine which box is blank
'Compute that missing value and put in text box
If txtDeposit.Text = "" Then
'Deposit missing
  Deposit = Final / (((1 + IntRate) ^ Months - 1) / IntRate)
  txtDeposit.Text = Format(Deposit, "#####0.00")
Elseif txtInterest.Text = "" Then
'Interest missing - requires iterative solution
  IntNew = (Final / (0.5* Months * Deposit) - 1) / Months
  Do
    IntRate = IntNew
    Fcn = (1 + IntRate) ^ Months - Final * IntRate / Deposit - 1
    FcnD = Months * (1 + IntRate) ^ (Months - 1) - Final / Deposit
    IntNew = IntRate - Fcn / FcnD
  Loop Until Abs(IntNew - IntRate) < 0.00001 / 12
  Interest = IntNew * 1200
  txtInterest.Text = Format(Interest, "##0.00")
Elseif txtMonths.Text = "" Then
'Months missing
  Months = Log(Final * IntRate / Deposit + 1) / Log(1 + IntRate)
  txtMonths.Text = Format(Months, "###.0")
Elseif txtFinal.Text = "" Then
'Final value missing
  Final = Deposit * ((1 + IntRate) ^ Months - 1) / IntRate
  txtFinal.Text = Format(Final, "#####0.00")
End If
End Sub
```

In this code, we first read the text information from all four text boxes and based on which one is blank, compute the missing information and display it in the corresponding text box. Solving for missing **Deposit**, **Months**, or **Final** information is a straightforward manipulation of the equation given in Example 2-2.

If the **Interest** value is missing, we have to solve an Mth-order polynomial using something called Newton-Raphson iteration - a good example of using a Do loop. Finding the **Interest** value is straightforward. What we do is guess at what the interest is, compute a better guess (using Newton-Raphson iteration), and repeat the process (loop) until the old guess and the new guess are close to each other. You can see each step in the code.

5. Test and save your application. Go home and relax.

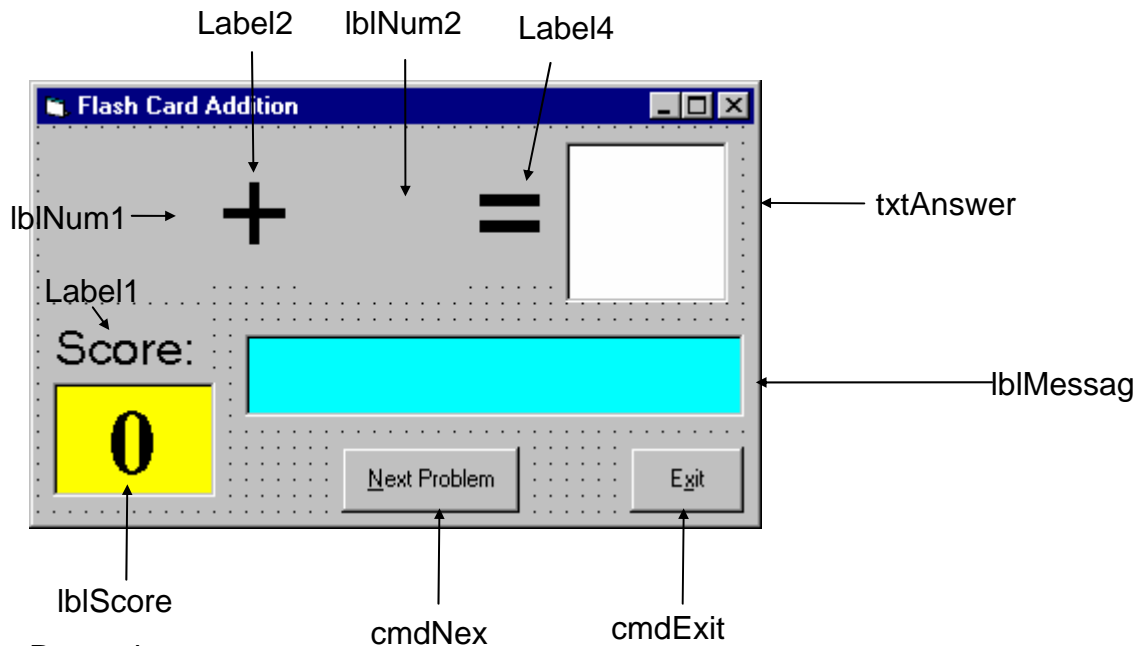
Exercise 2-1

Flash Card Addition Problems

Write an application that generates random addition problems. Provide some kind of feedback and scoring system as the problems are answered.

My Solution:

Form:



Properties:

Form **frmAdd**:

BorderStyle = 1 - Fixed Single
Caption = Flash Card Addition

CommandButton **cmdNext**:

Caption = &Next Problem
Enabled = False

CommandButton **cmdExit**:

Caption = E&xit

TextBox **txtAnswer**:

FontName = Arial
FontSize = 48
MaxLength = 2

Label **lblMessage:**

Alignment = 2 - Center
BackColor = &H00FFFF00& (Cyan)
BorderStyle = 1 - Fixed Single
FontName = MS Sans Serif
FontBold = True
FontSize = 24
FontItalic = True

Label **lblScore:**

Alignment = 2 - Center
BackColor = &H0000FFFF& (Yellow)
BorderStyle = 1 - Fixed Single
Caption = 0
FontName = Times New Roman
FontBold = True
FontSize = 36

Label **Label1:**

Alignment = 2 - Center
Caption = Score:
FontName = MS Sans Serif
FontSize = 18

Label **Label4:**

Alignment = 2 - Center
Caption = =
FontName = Arial
FontSize = 48

Label **lblNum2:**

Alignment = 2 - Center
FontName = Arial
FontSize = 48

Label **Label2:**

Alignment = 2 - Center
Caption = +
FontName = Arial
FontSize = 48

Label **lblNum1:**

Alignment = 2 - Center
FontName = Arial
FontSize = 48

Code:

General Declarations:

```
Option Explicit
Dim Sum As Integer
Dim NumProb As Integer, NumRight As Integer
```

cmdExit Click Event:

```
Private Sub cmdExit_Click()
End
End Sub
```

cmdNext Click Event:

```
Private Sub cmdNext_Click()
'Generate next addition problem
Dim Number1 As Integer
Dim Number2 As Integer
txtAnswer.Text = ""
lblMessage.Caption = ""
NumProb = NumProb + 1
'Generate random numbers for addends
Number1 = Int(Rnd * 21)
Number2 = Int(Rnd * 21)
lblNum1.Caption = Format(Number1, "#0")
lblNum2.Caption = Format(Number2, "#0")
'Find sum
Sum = Number1 + Number2
cmdNext.Enabled = False
txtAnswer.SetFocus
End Sub
```

Form Activate Event:

```
Private Sub Form_Activate()
Call cmdNext_Click
End Sub
```

Form Load Event:

```
Private Sub Form_Load()  
Randomize Timer  
NumProb = 0  
NumRight = 0  
End Sub
```

txtAnswer KeyPress Event:

```
Private Sub txtAnswer_KeyPress(KeyAscii As Integer)  
Dim Ans As Integer  
'Check for number only input and for return key  
If (KeyAscii >= vbKey0 And KeyAscii <= vbKey9) Or KeyAscii = vbKeyBack Then  
Exit Sub  
ElseIf KeyAscii = vbKeyReturn Then  
'Check answer  
Ans = Val(txtAnswer.Text)  
If Ans = Sum Then  
NumRight = NumRight + 1  
lblMessage.Caption = "That's correct!"  
Else  
lblMessage.Caption = "Answer is " + Format(Sum, "#0")  
End If  
lblScore.Caption = Format(100 * NumRight / NumProb, "##0")  
cmdNext.Enabled = True  
cmdNext.SetFocus  
Else  
KeyAscii = 0  
End If  
End Sub
```

Visual Basic Functions

Visual Basic offers a rich assortment of built-in **functions**. The on-line help utility will give you information on any or all of these functions and their use. Some examples are:

Function	Value Returned
Abs	Absolute value of a number
Asc	ASCII or ANSI code of a character
Chr	Character corresponding to a given ASCII or ANSI code
Cos	Cosine of an angle
Date	Current date as a text string
Format	Date or number converted to a text string
Left	Selected left side of a text string
Len	Number of characters in a text string
Mid	Selected portion of a text string
Now	Current time and date
Right	Selected right end of a text string

Rnd	Random number
Sin	Sine of an angle
Sqr	Square root of a number
Str	Number converted to a text string
Time	Current time as a text string
Timer	Number of seconds elapsed since midnight
Val	Numeric value of a given text string

Example 1-3

Stopwatch Application - Attaching Code

All that's left to do is attach code to the application. We write code for every event a response is needed for. In this application, there are three such events: clicking on each of the command buttons.

1. Double-click anywhere on the form to open the code window. Or, select 'View Code' from the project window.
2. Click the down arrow in the Object box and select the object named **(general)**. The Procedure box will show **(declarations)**. Here, you declare three form level variables:

Option Explicit

Dim StartTime As Variant

Dim EndTime As Variant

Dim ElapsedTime As Variant

The **Option Explicit** statement forces us to declare all variables. The other lines establish **StartTime**, **EndTime**, and **ElapsedTime** as variables global within the form.

3. Select the **cmdStart** object in the Object box. If the procedure that appears is not the Click procedure, choose **Click** from the procedure box. Type the following code which begins the timing procedure. Note the **Sub** and **End Sub** statements are provided for you:

Sub cmdStart_Click ()

'Establish and print starting time

StartTime = Now

lblStart.Caption = Format(StartTime, "hh:mm:ss")

lblEnd.Caption = ""

lblElapsed.Caption = ""

End Sub

In this procedure, once the **Start Timing** button is clicked, we read the current time and print it in a label box. We also blank out the other label boxes. In the code above (and in all code in these notes), any line beginning with a single quote (') is a comment. You decide whether you want to type these lines or not. They are not needed for proper application operation.

4. Now, code the **cmdEnd** button.

```
Sub cmdEnd_Click ()  
'Find the ending time, compute the elapsed time  
'Put both values in label boxes  
EndTime = Now  
ElapsedTime = EndTime - StartTime  
lblEnd.Caption = Format(EndTime, "hh:mm:ss")  
lblElapsed.Caption = Format(ElapsedTime, "hh:mm:ss")  
End Sub
```

Here, when the **End Timing** button is clicked, we read the current time (**End Time**), compute the elapsed time, and put both values in their corresponding label boxes.

5. And, finally the **cmdExit** button.

```
Sub cmdExit_Click ()  
End  
End Sub
```

This routine simply ends the application once the **Exit** button is clicked.

6. Did you notice that as you typed in the code, Visual Basic does automatic syntax checking on what you type (if you made any mistakes, that is)?
7. Run your application by clicking the **Run** button on the toolbar, or by pressing <f5>. Pretty easy, wasn't it?
8. Save your application - see the **Primer** on the next page. Use the **Save Project As** option under the **File** menu. Make sure you save both the form and the project files.

If you have the time, some other things you may try with the Stopwatch Application:

- A. Try changing the form color and the fonts used in the label boxes and command buttons.
- B. Notice you can press the 'End Timing' button before the 'Start Timing' button. This shouldn't be so. Change the application so you can't do this. And make it such that you can't press the 'Start Timing' until 'End Timing' has been pressed. Hint: Look at the command button **Enabled** property.
- C. Can you think of how you can continuously display the 'End Time' and 'Elapsed Time'? This is a little tricky because of the event-driven nature of Visual Basic. Look at the **Timer** tool. Ask me for help on this one.

Quick Primer on Saving Visual Basic Applications:

When saving Visual Basic applications, you need to be concerned with saving both the forms (.FRM) and modules (.BAS) and the project file (.VBP). In either case, make sure you are saving in the desired directory. The current directory is always displayed in the Save window. Use standard Windows techniques to change the current directory.

There are four **Save** commands available under the **File** menu in Visual Basic:

Save [Form Name]	Save the currently selected form or module with the current name. The selected file is identified in the Project window.
Save [Form Name] As	Like Save File, however you have the option to change the file name
Save Project	Saves all forms and modules in the current project using their current names and also saves the project file.
Save Project As	Like Save Project, however you have the option to change file names. When you choose this option, if you have not saved your forms or modules, you will also be prompted to save those files. I always use this for new projects.

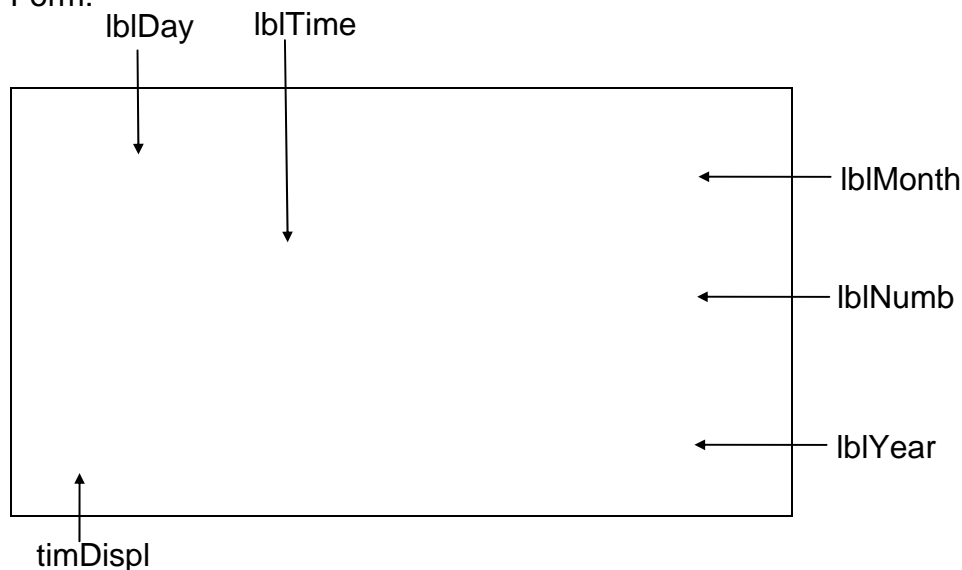
Exercise 1

Calendar/Time Display

Design a window that displays the current month, day, and year. Also, display the current time, updating it every second (look into the **Timer** control). Make the window look something like a calendar page. Play with object properties to make it pretty.

My Solution:

Form:



Properties:

Form **frmCalendar:**

Caption = My Calendar
BorderStyle = 1 - Fixed Single

Timer **timDisplay:**

Interval = 1000

Label **lblDay:**

Caption = Sunday
FontName = Times New Roman
FontBold = True
FontSize = 24

Label **lblTime:**

Caption = 00:00:00 PM
FontName = Times New Roman
FontBold = True
FontSize = 24

Label **lblYear:**

Alignment = 2 - Center
Caption = 1998
FontName = Times New Roman
FontBold = True
FontSize = 24

Label **lblNumber:**

Alignment = 2 - Center
Caption = 31
FontName = Arial
FontBold = True
FontSize = 72

Label **lblMonth:**

Alignment = 2 - Center
Caption = March
FontName = Times New Roman
FontBold = True
FontSize = 24

Code:

General Declarations:

Option Explicit

timDisplay Timer Event:

```
Private Sub timDisplay_Timer()  
Dim Today As Variant  
Today = Now  
lblDay.Caption = Format(Today, "dddd")  
lblMonth.Caption = Format(Today, "mmmm")  
lblYear.Caption = Format(Today, "yyyy")  
lblNumber.Caption = Format(Today, "d")  
lblTime.Caption = Format(Today, "h:mm:ss ampm")  
End Sub
```


Unit 6

Web Page Development

Fundamentals of HTML

Hypertext Markup Language, the coding language used to create hypertext documents for the World Wide Web. In HTML where a block of text can be surrounded with tags that indicate how it should appear (for example, in bold face or italics). Also, in HTML a word, a block of text, or an image can be linked to another file on the Web. HTML files are viewed with a World Wide Web browser.

This is a subset of Standard Generalized Markup Language (SGML). This language provides codes used to format hypertext documents. Individual codes are used to define the hierarchy and nature of various components of a document, as well as to specify hyperlinks.

HTML documents are written in plain text, but with the addition of tags, which describe or define the text they enclose. For example, the ANCHOR <A> tag placed around the hyperlinked text defines a link. It specifies the URL of the 'linked to' document, eg <A HREF="http://www. ...

Fundamentals of XML

Extensible Markup Language; defined by the World Wide Web Consortium (W3C) (2004b) as “a class of data objects called XML documents partially describes the behavior of computer programs which process them. XML documents are made up of storage units called entities, which contain either parsed or unparsed data. Parsed data is made up of characters, some of which form character data, and some of which form markup. ...

It is a W3C-recommended general-purpose markup language for creating special-purpose markup languages. It is a simplified subset of SGML, capable of describing many different kinds of data. Its primary purpose is to facilitate the sharing of data across different systems, particularly systems connected via the Internet.

Creating a Simple Web Page

To create a web page there are two methods

1. Create web page using a text editor
Eg: Note Pad
2. Create web page using standard package
Eg: Microsoft Front Page, Macromedia Dreamviewer and Adobe Fireworks

Create a web page using a text editor

```
<html>

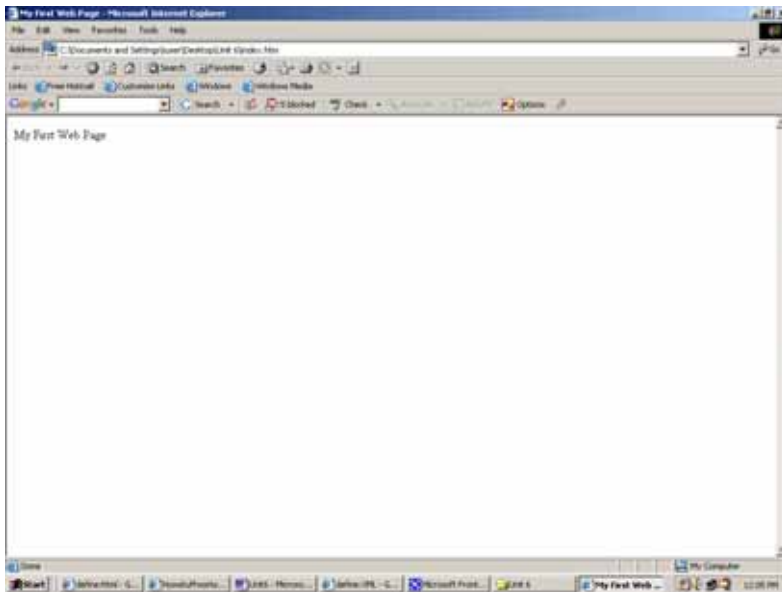
<head>
<title>New Page 1</title>
</head>

<body>

My First Web Page

</body>

</html>
```



Create a web page using standard package

For example, HTML code generated by FrontPage is as follows

```
<html>

<head>
<meta http-equiv="Content-Language" content="en-us">
<meta name="GENERATOR" content="Microsoft FrontPage 5.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<meta http-equiv="Content-Type" content="text/html; charset=windows-
1252">
<title>New Page 1</title>
</head>

<body>

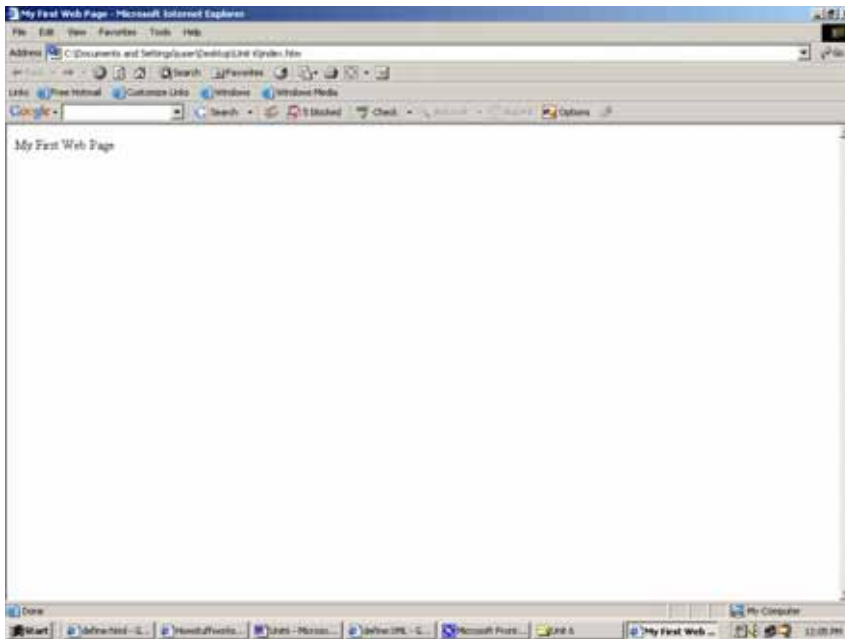
<p>My First Web Page</p>

</body>

</html>
```



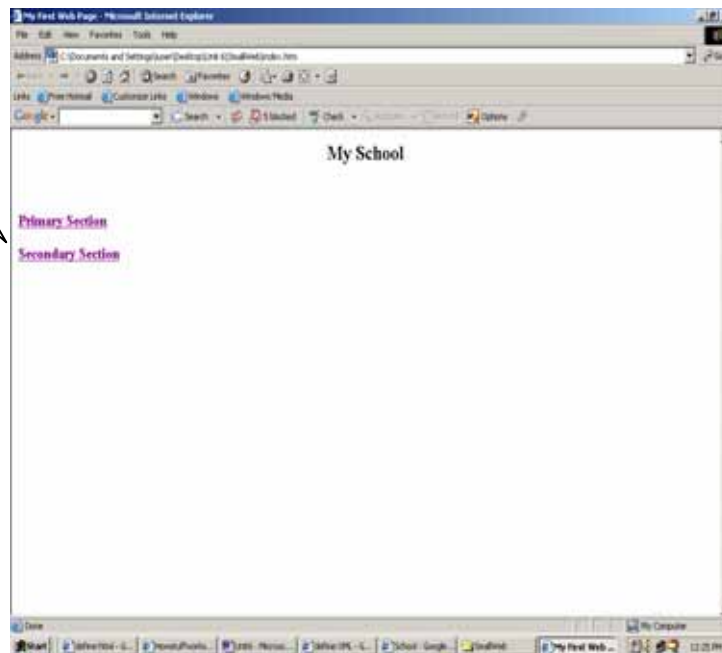
Preview of the desired web page on a Web Browser



Creating and linking multiple Pages

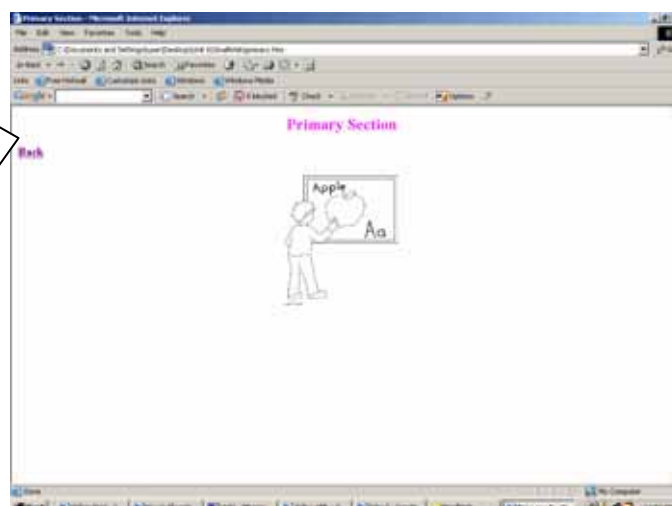
This explains how to create a small web site with multiple pages that has links along with them. First you have to create a web page called index. Most of the time index means the home page of the site you are going to create.

Create hyperlinks to primary and secondary pages here.

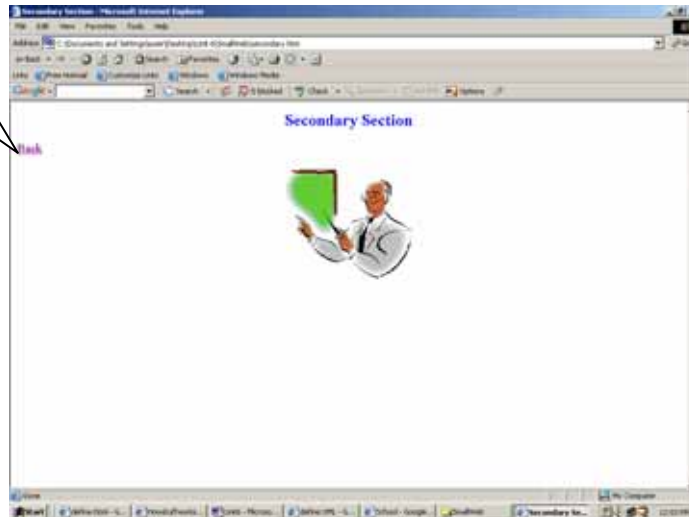


Next you have to create other pages. The two other pages available in this example are as follows.

Create hyperlink to index page here



Create
hyperlink to
index page
here



After creating other pages create hyperlinks from index page to primary and secondary pages.

```
<html>
```

```
<head>
```

```
<title>My First Web Page</title>
```

```
</head>
```

```
<body>
```

```
<p align="center"><b><font size="5">My School</font></b></p>
```

```
<p align="left">&nbsp;</p>
```

```
<p align="left"><font size="4"><b><a href="primary.htm">Primary  
Section</a></b></font></p>
```

```
<p align="left"><font size="4"><b><a href="secondary.htm">Secondary  
Section</a></b></font></p>
```

```
</body>
```

```
</html>
```

```
<html>
```

```
<head>
```

```
<title>Primary Section</title>
```

```
</head>
```

```
<body>
```

```
<p align="center"><b><font size="5" color="#FF00FF">Primary  
Section</font></b></p>
```

```
<p align="left"><b><font size="4"><a href="index.htm">Back</a></font></b></p>
```

```
<p align="center"></p>
```

```
</body>
```

```
</html>
```

```
<html>
```

```
<head>
<title>Secondary Section</title>
</head>
```

```
<body>
```

```
<p align="center"><b><font size="5" color="#0000FF">Secondary Section</font></b></p>
```

```
<p align="left"><font size="4" color="#0000FF"><a href="index.htm">Back</a></font></p>
```

```
<p align="center"></p>
```

```
</body>
```

```
</html>
```

Using HTML codes in a text editor is the Basic method to create a web page. The concept of HTML codes and tags is very important in this method. Let us discuss HTML Codes.

Hyper Text Markup Language (HTML)

What is an HTML File?

- HTML stands for Hyper Text Markup Language
- An HTML file is a text file containing small markup tags
- The markup tags tell the Web browser how to display the page
- An HTML file must have an htm or html file extension better to use html extension.

An HTML file can be created using a simple text editor or any text editor can be used.

Open notepad and Type the following text:

```
<html>
<head>
<title>Title of page</title>
</head>
<body>
This is my first homepage. <b>This text is in bold</b>
</body>
</html>
```

Output

This is my first homepage. This text is in bold

Save the file as "myfirst.htm".

Start Internet browser and locate the HTML file you just created - "myfirst.htm" - select same and click "Open " (or "Open Page") in the File menu of your browser. A dialog box will appear. Select "Browse" (or "Choose File"). Now you should see an address in the dialog box, for example "C:\MyDocuments\myfirst.htm". Click OK. The page will be displayed.

Example Explained

- The first tag <html> tells your browser that this is the start of an HTML document. The last tag </html> tells your browser that this is the end of the HTML document.
- The text between the <head> tag and the </head> tag is header information. Header information is not displayed in the browser window.
- The text between the <title> tags is the title of your document. The title is displayed in your browser's caption.
- The text between the <body> tags is the text that will be displayed in your browser.
- The text between the and tags will be displayed in a bold font.

HTML elements are defined using HTML tags.

HTML Tags

HTML tags are used to mark-up HTML elements

HTML tags are surrounded by the two angle brackets < and >

HTML tags normally come in pairs like and

The first tag in a pair is the start tag, the second tag is the end tag

The text between the start and end tags is the element content

HTML tags are not case sensitive, means the same as

HTML Elements

Recall the HTML example from the previous page:

```
<html>
<head><
title>Title of page</title>
</head>
<body>This is my first homepage. <b>This text is in bold</b>
</body>
</html>
```

This is an HTML element:

```
<b>This text is in bold</b>
```

The HTML element between start tag: and end tag: defines an HTML element that should be displayed in bold.

This is also an HTML element:

```
<body>
This is my first homepage. <b>This text is in bold</b>
</body>
```

This HTML element between start tag `<body>` and end tag `</body>` defines the HTML element that contains the body of the HTML document.

Tag Attributes

Tags can have attributes. Attributes can provide additional information about the HTML elements on your page.

This tag defines the body element of your HTML page: `<body>`. With an added `bgcolor` attribute, tells the browser that the background color of your page should be red, like this: `<body bgcolor="red">`.

This tag defines an HTML table: `<table>`. With an added `border` attribute, you can tell the browser that the table should have no borders: `<table border="0">`

Attributes always come in name/value pairs like this: `name="value"`.

Attributes are always added to the start tag of an HTML element.

Quote Styles, "red" or 'red'?

Attribute values should always be enclosed in quotes. Double style quotes are the most common, but single style quotes are also allowed.

In some rare situations, like when the attribute value itself contains quotes, it is necessary to use single quotes:

```
name='Nimal "Shot" Fernando'
```

The most important tags in HTML are tags that define headings, paragraphs and line breaks.

Headings

Headings are defined with the `<h1>` to `<h6>` tags. `<h1>` defines the largest heading. `<h6>` defines the smallest heading.

```
<h1>This is a heading</h1>
```

```
<h2>This is a heading</h2>
```

```
<h3>This is a heading</h3>
```

```
<h4>This is a heading</h4>
```

```
h5>This is a heading</h5>
```

```
<h6>This is a heading</h6>
```

HTML automatically adds an extra blank line before and after a heading.

Output

This is a heading

This is a heading

This is a heading

This is a heading

This is a heading

This is a heading

Paragraphs

Paragraphs are defined with the `<p>` tag.

```
<p>This is a paragraph</p>
```

```
<p>This is another paragraph</p>
```

HTML automatically adds an extra blank line before and after a paragraph.

Line Breaks

The
 tag is used to end a line, but there is no need to start a new paragraph. The
 tag forces a line break wherever you place it.

```
<p>This <br> is a Green<br>Color pen with line breaks</p>
```

The
 tag is an empty tag. It has no closing tag.

Output

This
is a Green
Color pen with line breaks

Comments in HTML

The comment tag is used to insert a comment in the HTML source code. A comment will be ignored by the browser. You can use comments to explain your code. The comments can help you when you edit the source code at a later date.

```
<!-- This is a comment -->
```

Note that you need an exclamation point after the opening bracket, but not before the closing bracket.

Basic HTML Tags

Tag	Description
<html>	Defines an HTML document
<body>	Defines the document's body
<h1> to <h6>	Defines header 1 to header 6
<p>	Defines a paragraph
 	Inserts a single line break
<hr>	Defines a horizontal rule
<!-->	Defines a comment

HTML Text Formatting

HTML defines a lot of elements for formatting output, like bold or italic text.

Below are a lot of examples that you can try out yourself:

Text formatting

```
<html>
<body>
  <b>This text is bold</b><br>
    <strong>
      This text is strong
    </strong>
  <br>
    <big>
      This text is big
    </big>
  <br>
</body>
</html>
```

Output

This text is bold

This text is strong

This text is big

HTML Character Entities

Some characters like the < character, have a special meaning in HTML, and therefore cannot be used in the text.

To display a less than sign (<) in HTML, we have to use a character entity.

Character Entities

If some special characters are to be displayed, insert character entities in the HTML source.

A character entity has three parts:

1. An ampersand (&)
2. An entity name or a # and an entity number
3. And finally a semicolon (;)

To display a less than sign in an HTML document write: < or <

Non-breaking Space

The most common character entity in HTML is the non-breaking space.

Normally HTML will truncate spaces in your text. If you write 10 spaces in your text HTML will remove 9 of them. To add spaces to your text, use the character entity.

The Most Common Character Entities:

Result	Description	Entity Name	Entity Number
	non-breaking space	 	
<	less than	<	<
>	greater than	>	>
&	ampersand	&	&
"	quotation mark	"	"
'	apostrophe	' (does not work in IE)	'

Some Other Commonly Used Character Entities:

Result	Description	Entity Name	Entity Number
¢	cent	¢	¢
£	pound	£	£
¥	yen	¥	¥
§	section	§	§
©	copyright	©	©
®	registered trademark	®	®
x	multiplication	×	×
÷	division	÷	÷

HTML Links

HTML uses a hyperlink to link to another document on the Web.

The Anchor Tag and the Href Attribute

HTML uses the <a> (anchor) tag to create a link to another document.

An anchor can point to any resource on the Web: an HTML page, an image, a sound file, a movie, etc.

The syntax of creating an anchor:

```
<a href="url">Text to be displayed</a>
```

The <a> tag is used to create an anchor to link from, the href attribute is used to address the document to link to, and the words between the “open and close” of the anchor tag will be displayed as a hyperlink.

This anchor defines a link to Department of Examination:

```
<a href="http://www.doenets.lk/">Visit Department of Examination ! </a>
```

The line above will look like this in a browser:

The Target Attribute

With the target attribute, you can define where the linked document will be opened.

The line below will open the document in a new browser window:

```
<a href=http://www.doenets.lk /  
target="_blank"> Visit Department of Examination!</a>
```

The Anchor Tag and the Name Attribute

The name attribute is used to jump directly into a specific section on a page without scrolling.

Below is the syntax of a named anchor:

```
<a name="label">Text to be displayed</a>
```

The name “attribute” is used to create a named anchor. The name of the anchor can be any text you care to use.

The line below defines a named anchor:

```
<a name="lessons">Select a lesson</a>
```

Notice that a named anchor is not displayed in a special way.

To link directly to the "lessons" section, add a # sign and the name of the anchor to the end of a URL, like this:

```
<a href="http:// http://www.doenets.lk //html_links.asp#results">  
Jump to the results Section</a>
```

A hyperlink to the Useful Tips Section from WITHIN the file "html_links.asp" will look like this:

```
<a href="#lessons">Jump to the lessons Section</a>
```

HTML Frames

Frames

With frames, more than one HTML document in can be displayed in the same browser window.

Each HTML document is called a frame, and each frame is independent of the others.

The disadvantages of using frames are:

- The web developer must keep track of more HTML documents
- It is difficult to print the entire page

The Frameset Tag

- The <frameset> tag defines how to divide the window into frames
- Each frameset defines a set of rows or columns
- The values of the rows/columns indicate the amount of screen area each row/column will occupy

The Frame Tag

- The <frame> tag defines what HTML document to put into each frame

In the example below we have a frameset with two columns. The first column is set to 25% of the width of the browser window. The second column is set to 75% of the width of the browser window. The HTML document "frame_a.htm" is put into the first column, and the HTML document "frame_b.htm" is put into the second column:

```
<frameset cols="25%,75%">
  <frame src="frame_a.htm">
  <frame src="frame_b.htm">
</frameset>
```

If frame borders are visible, the user can resize it by dragging the border. Resizing can be prevented by adding noresize="noresize" to the <frame> tag.

Add the <noframes> tag for browsers to avoid frames.

Important: You cannot use the <body></body> tags together with the <frameset></frameset> tags! However, if you add a <noframes> tag containing some text for browsers that do not support frames, you will have to enclose the text in <body></body> tags! See how it is done in the first example below.

Navigation frame

The navigation frame contains a list of links with the second frame as the target. The file called "Frames.html" contains three links. The source code of the links:

```
<a href="frame_a.htm" target="showframe">Frame a</a><br>
<a href="frame_b.htm" target="showframe">Frame b</a><br>
<a href="frame_c.htm" target="showframe">Frame c</a>
```

The second frame will show the linked document.

Frame Tags

Tag	Description
<frameset>	Defines a set of frames
<frame>	Defines a sub window (a frame)
<noframes>	Defines a noframe section for browsers that do not handle frames
<iframe>	Defines an inline sub window (frame)

The Image Tag and the Src Attribute

In HTML, images are defined with the tag.

The tag is empty, which means that it contains attributes only and it has no closing tag.

You need to use the src attribute to display an image on a page. Src stands for "source". The value of the src attribute is the URL of the image you want to display on your page.

The syntax of defining an image:

```

```

The URL points to the location where the image is stored. An image named "flag.gif" located in the directory "Picture" on "www. schoolsnet.com" has the URL: <http://www.schoolsnet.com/images/boat.gif>.

The browser puts the image where the image tag occurs in the document. If you put an image tag between two paragraphs, the browser shows the first paragraph, then the image, and then the second paragraph.

The Alt Attribute

The alt attribute is used to define an "alternate text" for an image. The value of the alt attribute is an author-defined text:

```

```

The "alt" attribute tells the reader what he or she is missing on a page if the browser can't load images. The browser will then display the alternate text instead of the image. It is a good practice to include the "alt" attribute for each image on a page, to improve the display and usefulness of your document for people who have text-only browsers.

Image Tags

Tag	Description
	Defines an image
<map>	Defines an image map
<area>	Defines a clickable area inside an image map

HTML Backgrounds

A good background can make a Web site look really great.

Backgrounds

The <body> tag has two attributes where you can specify backgrounds. The background can be a color or an image.

Bgcolor

The bgcolor attribute specifies a background-color for an HTML page. The value of this attribute can be a hexadecimal number, an RGB value, or a color name:

```
<body bgcolor="#000000">
```

```
<body bgcolor="rgb(0,0,0)">
```

```
<body bgcolor="black">
```

The lines above all set the background-color to black.

Background

The background attribute specifies a background-image for an HTML page. The value of this attribute is the URL of the image you want to use. If the image is smaller than the browser window, the image will repeat itself until it fills the entire browser window.

```
<body background="clouds.gif">
```

```
<body background="http://www.schoolnet.com/clouds.gif">
```

The URL can be relative (as in the first line above) or absolute (as in the second line above).

When using a background image, consider the following:

- Will the background image increase the loading time too much?
- Will the background image look good with other images on the page?
- Will the background image look good with the text colors on the page?
- Will the background image look good when it is repeated on the page?
- Will the background image take away the focus from the text?

HTML Lists

Unordered Lists

• An unordered list is a list of items. The list items are marked with bullets (typically small black circles).

• An unordered list starts with the tag. Each list item starts with the tag.

```
<ul>
```

```
<li>Coffee</li>
```

```
<li>Milk</li>
```

```
</ul>
```

Here is how it looks in a browser:

- Coffee
- Milk

Inside a list item you can put paragraphs, line breaks, images, links, other lists, etc.

Ordered Lists

An ordered list is also a list of items. The list items are marked with numbers.

An ordered list starts with the `` tag. Each list item starts with the `` tag.

```
<ol>
<li>Coffee</li>
<li>Milk</li>
</ol>
```

Here is how it looks in a browser:

- Coffee
- Milk

Inside a list item you can put paragraphs, line breaks, images, links, other lists, etc.

Definition Lists

A definition list is not a list of items. This is a list of terms and explanations of the terms.

A definition list starts with the `<dl>` tag. Each definition-list term starts with the `<dt>` tag. Each definition-list definition starts with the `<dd>` tag.

```
<dl><dt>Coffee</dt>
<dd>Black hot drink</dd>
<dt>Milk</dt><dd>
White cold drink</dd>
</dl>
```

Here is how it looks in a browser:

Coffee

Black hot drink

Milk

White cold drink

Inside a definition-list definition (the <dd> tag) you can put paragraphs, line breaks, images, links, other lists, etc.

List Tags

Tag	Description
	Defines an ordered list
	Defines an unordered list
	Defines a list item
<dl>	Defines a definition list
<dt>	Defines a definition term
<dd>	Defines a definition description
<dir>	Deprecated. Use instead
<menu>	Deprecated. Use instead

HTML Tables

With HTML you can create tables.

Tables

Tables are defined with the <table> tag. A table is divided into rows (with the <tr> tag), and each row is divided into data cells (with the <td> tag). The letters td stands for "table data," which is the content of a data cell. A data cell can contain text, images, lists, paragraphs, forms, horizontal rules, tables, etc.

```
<table border="1">
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr></table>
```

How it looks in a browser:

row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

Tables and the Border Attribute

To display a table with borders, to use the border attribute:

```
<table border="1">
<tr>
<td>Row 1, cell 1</td>
<td>Row 1, cell 2</td>
</tr>
</table>
```

row 1, cell 1	row 1, cell 2
row 2, cell 1	

Note that the borders around the empty table cell are missing (NB! Mozilla Firefox displays the border).

To avoid this, add a non-breaking space () to empty data cells, to make the borders visible:

```
<table border="1">
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
td>row 2, cell 1</td>
<td>&nbsp;</td>
</tr>
</table>
```

Headings in a Table

Headings in a table are defined with the <th> tag.

```
<table border="1">
<tr>
<th>Heading</th>
<th>Another Heading</th>
</tr>
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
```

How it looks in a browser:

Heading	Another Heading
row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

Empty Cells in a Table

Table cells with no content are not displayed very well in most browsers.

```
<table border="1"><tr><td>row 1, cell 1</td><td>row 1, cell 2</td></tr><tr><td>row 2, cell 1</td><td></td></tr></table>
```

How it looks in a browser:

row 1, cell 1	row 1, cell 2
row 2, cell 1	

Table Tags

Tag	Description
<table>	Defines a table
<th>	Defines a table header
<tr>	Defines a table row
<td>	Defines a table cell
<caption>	Defines a table caption
<colgroup>	Defines groups of table columns
<col>	Defines the attribute values for one or more columns in a table
<thead>	Defines a table head
<tbody>	Defines a table body
<tfoot>	Defines a table footer










HTML Colors

Colors are displayed combining RED, GREEN, and BLUE light sources.

Color Values

Colors are defined using a hexadecimal notation for the combination of Red, Green, and Blue color values (RGB). The lowest value that can be given to one light source is 0 (hex #00). The highest value is 255 (hex #FF).




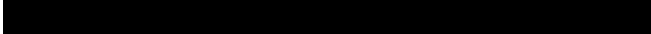



This table shows the result of combining Red, Green, and Blue light sources:

Color	Color HEX	Color RGB
	#000000	rgb(0,0,0)
	#FF0000	rgb(255,0,0)
	#00FF00	rgb(0,255,0)
	#0000FF	rgb(0,0,255)
	#FFFF00	rgb(255,255,0)
	#00FFFF	rgb(0,255,255)
	#FF00FF	rgb(255,0,255)
	#C0C0C0	rgb(192,192,192)
	#FFFFFF	rgb(255,255,255)

Color Names

Most browsers support a collection of color names.

Note: Only 16 color names are supported by the W3C HTML 4.0 standard (aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, and yellow). For all other colors you should use the Color HEX value.

Color	Color HEX	Color Name
	#F0F8FF	AliceBlue
	#FAEBD7	AntiqueWhite
	#7FFFD4	Aquamarine
	#000000	Black
	#0000FF	Blue
	#8A2BE2	BlueViolet
	#A52A2A	Brown

Web Development Tools

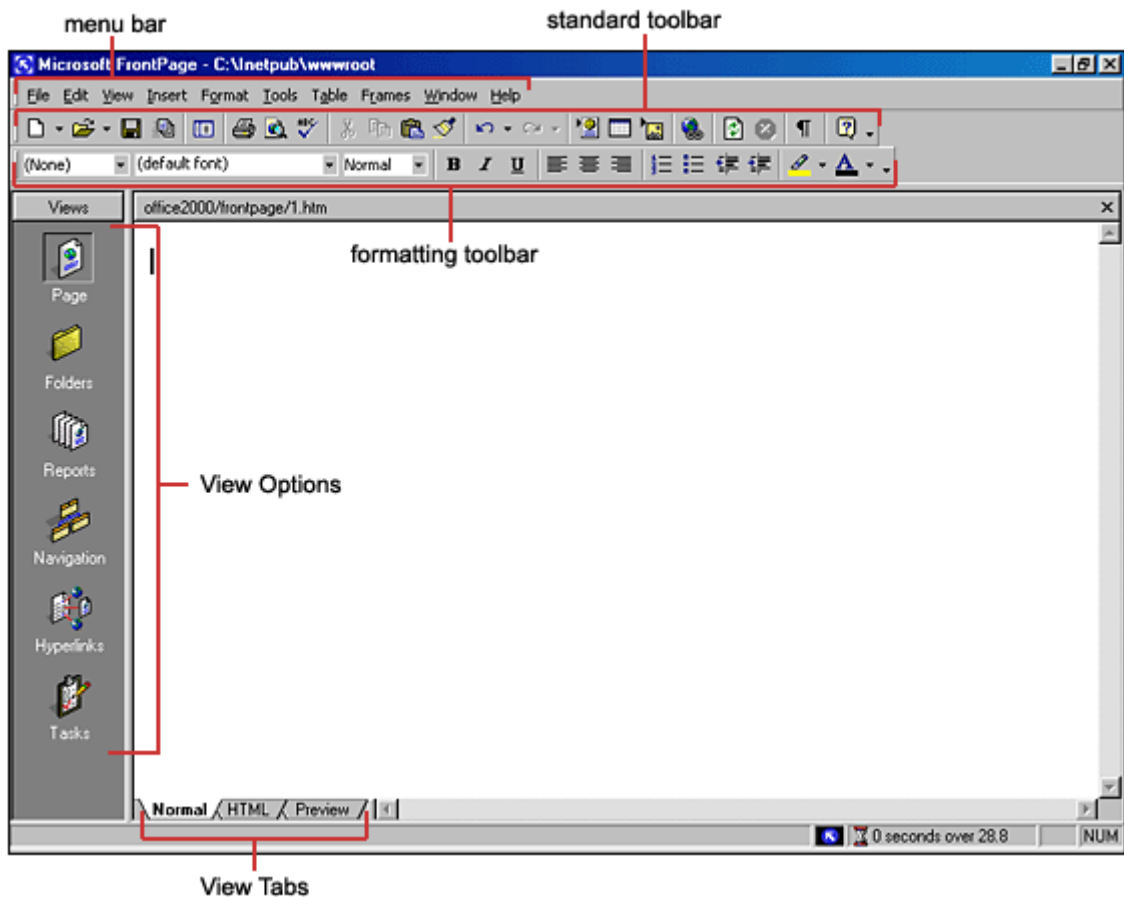
There are many tools used to develop web sites. Most popular tools are

Microsoft FrontPage
Macromedia Dreamviewer
Adobe Fireworks

In this book we are going to discuss two of the most popular web developing tools. First we are going to discuss the Microsoft FrontPage

FrontPage Screen Layout

Below is a diagram of the default page layout in FrontPage. You can change the view by selecting a different **View Option**.

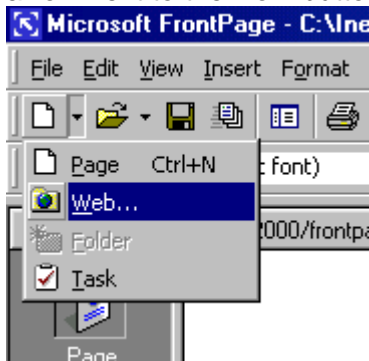


Views

- **Page view** gives you a WYSIWYG editing environment for creating and editing web pages.
- **Folders view** lists all of the files and folders in your web for easy management.
- **Reports view** identifies problems with pages and links in the web including slow-loading pages, broken links, and other errors.
- **Navigation view** lists the navigation order of the site and allows you to change the order that a user would view the pages.
- **Hyperlinks view** allows you to organize the links in the web pages.
- **Tasks view** provides a grid for inputting tasks you need to complete in your web.

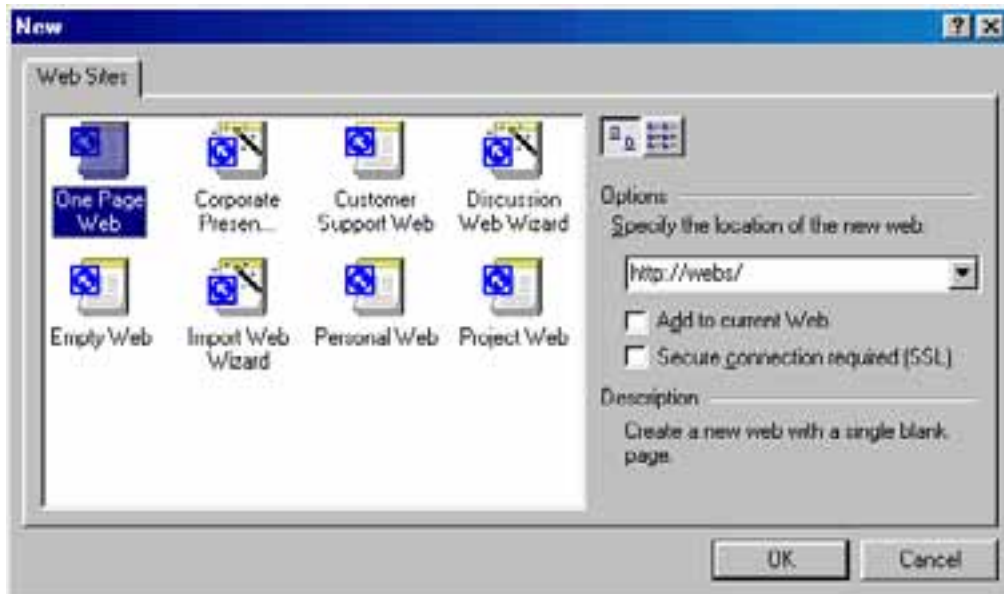
Creating a Web Using the Web Wizard

- Open FrontPage and select **File|New|Web...** from the menu bar or click the small down arrow next to the **New** button on the standard toolbar and select **Web....**



- Select the type of web you want to create. It is usually best to create a simple **One Page Web** which you can add additional blank pages to as you need them. Enter a location for the web in the box provided beginning with "http://". This is the location where you can preview

the web on your computer. It will need to be copied to the server to be viewed to the world on the WWW.

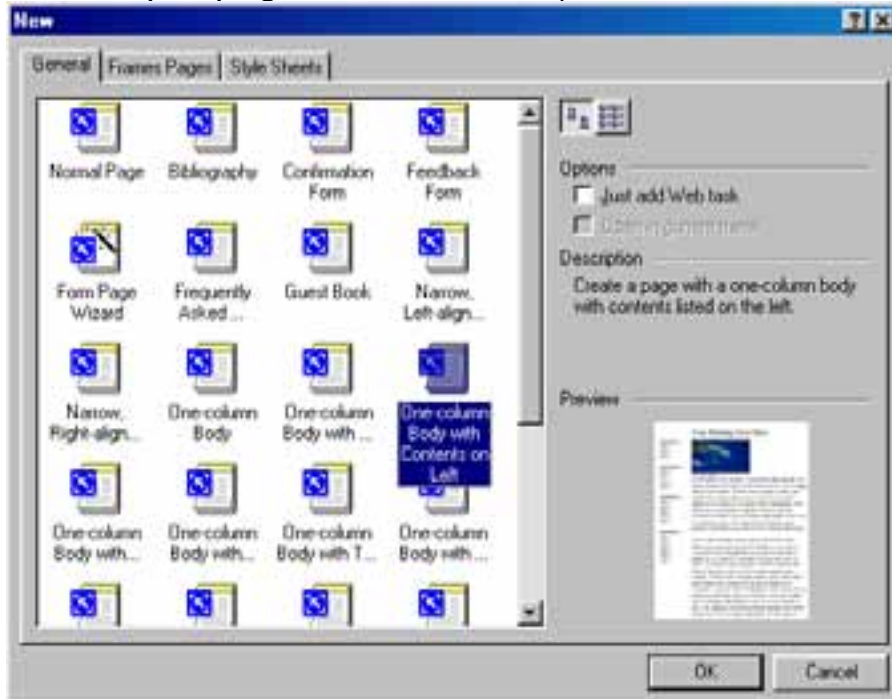


- Click **OK** and wait for FrontPage to finish creating the web.
- Now, explore your web. Click **Folders view** to see the initial page (default.htm) that was created and two folders. The "images" folder is where you will place all your graphics and photos. While it is not imperative that the images be placed in a separate folder, it keeps the web organized.
- Click on **Reports view** to see a list of reports for the site. As you construct your web, this page will be much more useful. From here, you can identify and correct broken hyperlinks and fix large pages that take a long time to load.
- View the navigation layout of the web by clicking **Navigation view**. Right now, there is only one page - the home page - listed. As more pages are added, this page becomes helpful to see how all your pages are linked together.
- Hyperlinks view allows you to manage the links on your pages.
- Optional - in Tasks view, list the tasks that need to be accomplished to create the web. Select **Edit|Task|Add Tasks** to add a task. Or click the down arrow beside the New button on the standard toolbar.
- Make pages and save them, marking them as completed in the task view.
- Click **Folders view** to locate the open the next page to work on.
- When you are ready to publish your web on the FGCU server, copy the folder to the server.

Creating a Web Page from a Template

FrontPage provides many individual page templates that can be added to any web. Follow these steps to add a template to a web page.

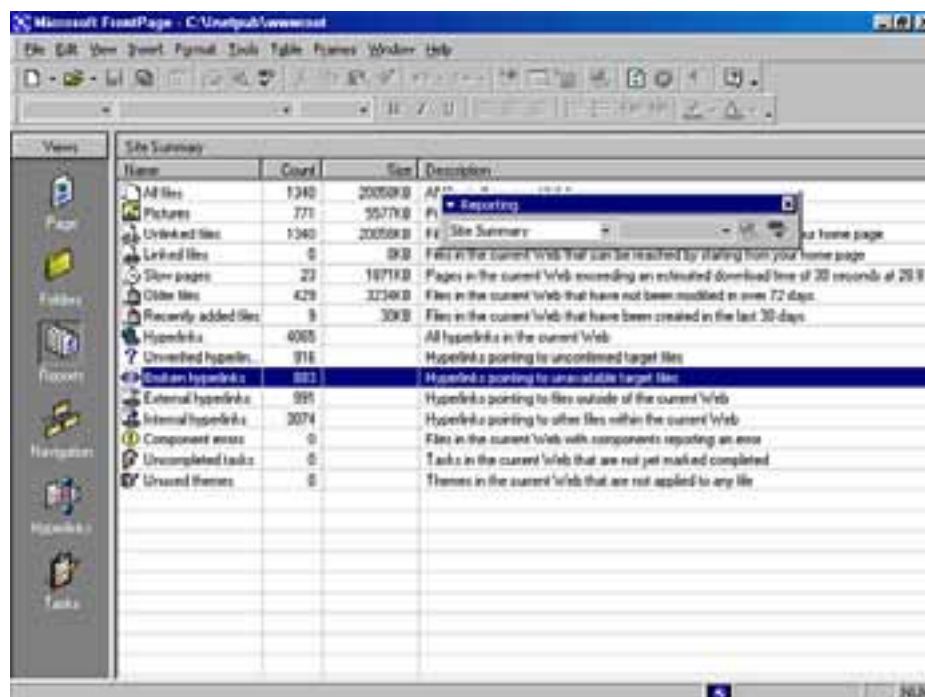
- Select **File|New|Page...** and choose a template.



- Select a template and click **OK**.
- Replace the place-holding body text with your own text and photos with images you would like on your web page.

Report View

When your web is completed, click **Reports** view to verify that links are correct and use the **Reporting** toolbar to switch between reports.



Open A Web

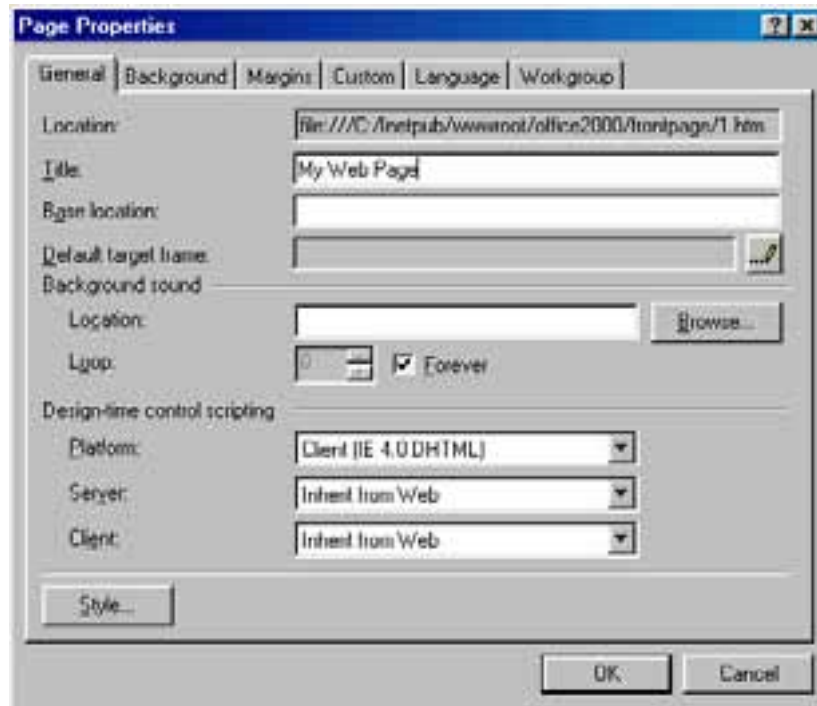
To open a web you have already created, select **File|Open Web...** from the menu bar. Select the web folder from the list and click **Open**.

Saving A Web

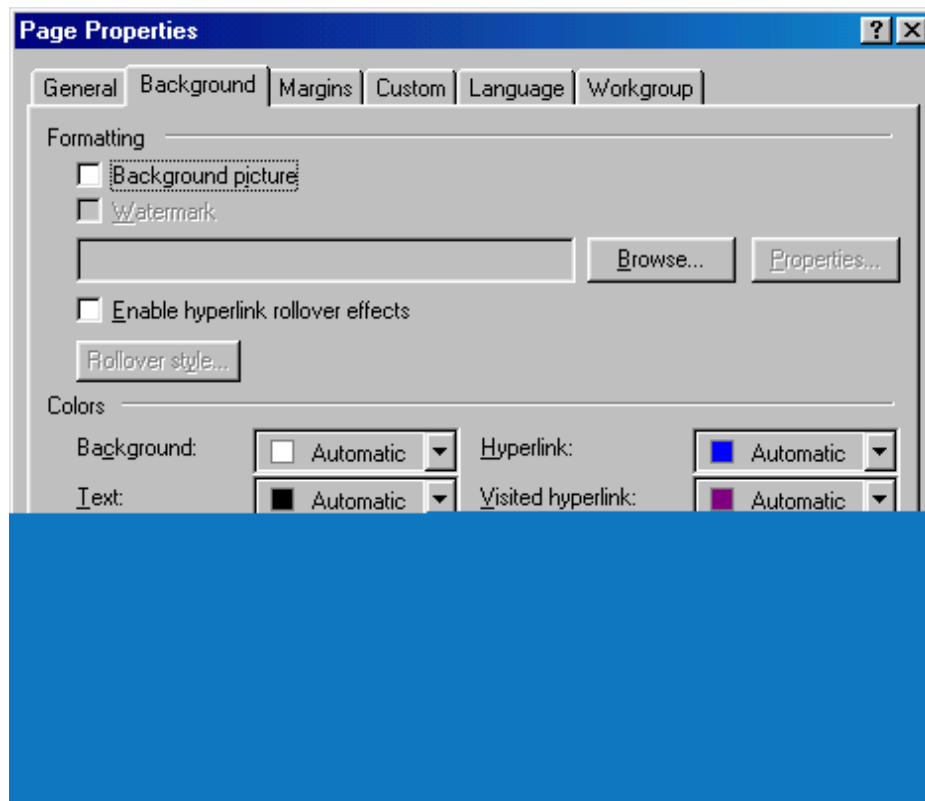
Save all the pages within the web created by FrontPage. These pages, however, are not visible to anyone on the Internet. You must copy the entire web folder to a network drive.

Page Properties

Change various page properties by selecting **File|Properties** from the menu bar. The Page Properties window will allow you to change many general properties, the page background, margins, and more.



- **General** - Under the General tab, one property that needs to be changed is the **Title**. This is the text that will appear across the top of the screen above the browser's menu bar when the page is viewed on the web. Background sounds are not recommended and design-time control scripting options do not need to be changed.



- **Background -**

Check the **Background picture** box and select an image by clicking the **Browse...** button to add a repeating graphic to the background of the page.

Enable hyperlink rollover effects adds a [Cascading Style Sheet](#) to the page that causes the appearance of text links to change when the mouse is placed over them. These effects are not visible in Netscape version 4 and lower.

Set the **Background** color and a default **Text** color if it is not black.

Hyperlink colors can be changed as well. The color set for **Hyperlink** will be the color of the text of a link that has not been viewed yet by the web site user. **Visited hyperlink** is the color the link will turn after the page has been visited. **Active hyperlink** is the color of the link as it is being pressed. This color is usually barely seen as the user quickly clicks the link. The default colors that web users are used to are blue for normal and purple for visited. Refrain from swapping these colors so users will not be confused!

- **Margins** - Set the top and left margin width by pixels if necessary. It is not necessary to alter any of the properties on the remaining tabs.

Themes

Themes can quickly add color, graphics, and a common layout to your web pages.

- Open a web page and select **Format|Theme** from the menu bar or right-click on the page and select **Theme...** from the shortcut menu.



- Under **Apply Theme to**, select **All pages** to add the theme to all pages in your web or **Selected page(s)** to only apply the theme to activated pages.
- Scroll through the theme selections and highlight the theme names to preview the theme in the **Sample of Theme** window. Click the check boxes to change the theme as well.
 - **Vivid Colors** enhances the colors of the theme.
 - **Active Graphics** will convert navigation buttons to Java applets that change when the mouse hovers over them.
 - **Background picture** including a repeating background image to the page. Uncheck the box for a plain color background.
 - **Apply Using CSS** will add the properties to a style sheet.
- Click **OK** when you have chosen the theme

Removing a Theme

To remove a theme from a page after it has been applied, select **Format|Theme** from the menu bar and select the first "(no theme)" option from the themes list. Click **OK**.

Font Properties

Many properties of fonts can be changed from the **Font** dialog box. Highlight the text that will be formatted and select **Format|Font** from the menu bar.



- **Font** - Select a simple, common font for the web page. Keep in mind that the list that appears in FrontPage is the list of fonts loaded onto *your* computer while many of the visitors of your web site will not have the same fonts. Choose a font such as Arial, Geneva, Verdana, Helvetica, or another sans-serif font that is easy to read and most people have loaded on their computers.
- **Font Style** - Select bold, italics, or a combination of both.
- **Size** - Font sizes on web pages are designated by different values than the point sizes you may be used to working
- in Word and other word processing programs. Font sizes are listed in parentheses next to the HTML point sizes. A point size of 2 or 3 is usually best for paragraph text. Below are examples of the font sizes using Arial font.

font size 1
font size 2
font size 3
font size 4
font size 5

- **Effects** - Many of these effects are unnecessary and some are not viewable on all browsers. It is not recommended that you **underline** any text as this will confuse your user since links are usually underlined. Use bold and italics to emphasize text instead of underlining. **Blink** is an old HTML specification and since it is quite annoying, many browsers no longer support it. **Strong** and **Emphasis** produce similar results to bold and italics.
- Press **OK** when finished.

Headings

Explain these. They generally don't work properly in Netscape, but do work when assigned values in themes.

Converting Text to Tables

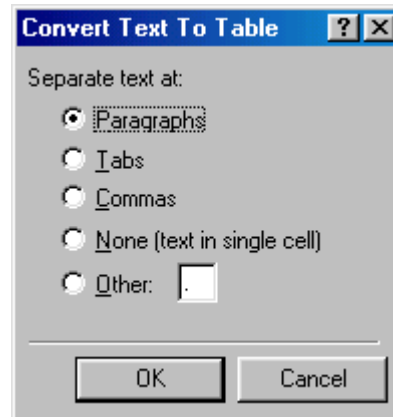
The text below was typed into FrontPage by pressing the **TAB** key after each number and **ENTER** to begin each new line.

FGCU Microsoft Office Tutorials

1 Microsoft FrontPage tutorial

2 Microsoft PowerPoint tutorial

The text can be put into a table by selecting **Table|Convert|Text to Table** from the menu bar. This dialog window will appear. Make a selection and click **OK**.



- **Paragraphs** - A new table row will begin at each new paragraph. Each line is placed in a single cell on a new row.

FGCU Microsoft Office Tutorials
1 Microsoft FrontPage tutorial
2 Microsoft PowerPoint tutorial

- **Tabs** - A new column will begin at each tab stop.

FGCU Microsoft Office Tutorials	
1	Microsoft FrontPage tutorial
2	Microsoft PowerPoint tutorial

- **Commas** - A new column will begin at each comma. The text below produces the same table format as the TAB setup.

FGCU Microsoft Office Tutorials

1,Microsoft FrontPage tutorial

2,Microsoft PowerPoint tutorial

- **None** - All the highlight text will be placed into a single table cell.

FGCU Microsoft Office Tutorials
1 Microsoft FrontPage tutorial
2 Microsoft PowerPoint tutorial

- **Other** - Select another delimiter for creating a table.

Creating Links

Hyperlinks are text or graphics that can be clicked to bring the user to another web file such as a web page or graphic. They are the essence of the World Wide Web as they link pages within sites and web sites to other web sites. To create a hyperlink in FrontPage, follow these steps:

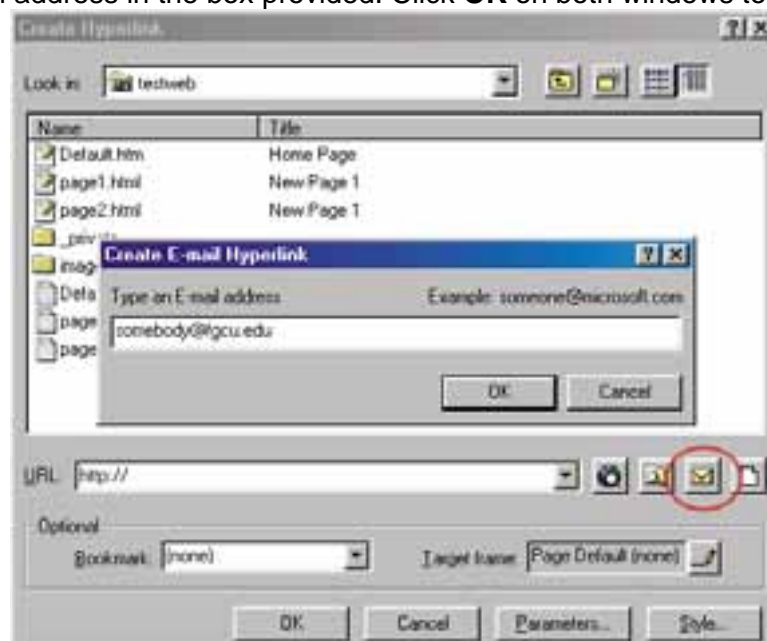
- Highlight the text or graphic that will be the hyperlink and select **Insert|Hyperlink** from the menu bar or pressing **CTRL+K**.



- If the link will lead to a page within your site, highlight the page on the list and click **OK**. If it is an external link that will lead to another web site, enter the URL in the **URL box**. External URLs **MUST** begin with "http://" or they will not work. For example, to link to the FGCU home page, type "http://www.fgcu.edu/" instead of "www.fgcu.edu/".

E-mail Links

Create an e-mail address link by highlighting the text (which should be written as the e-mail address) and pressing **CTRL+K**. Click the e-mail button with the envelope icon (circled in red below) and enter the e-mail address in the box provided. Click **OK** on both windows to finish.



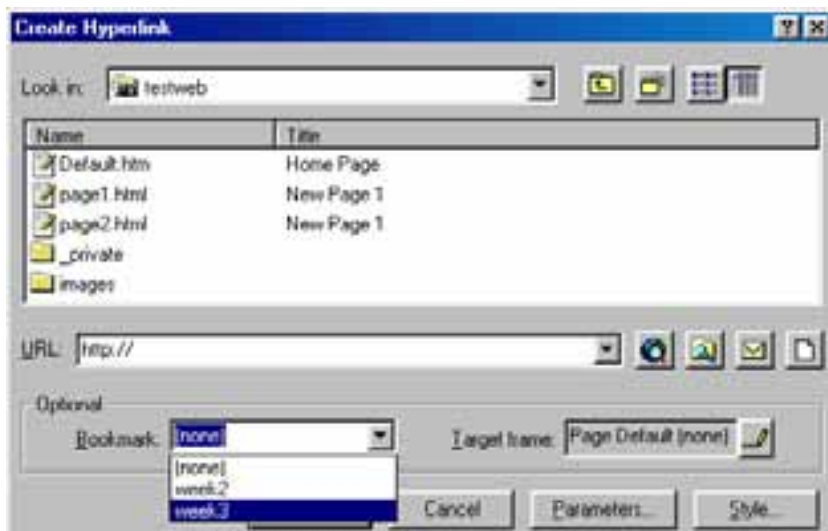
Bookmarks

Text and graphics can be set as bookmarks (called "anchors" everywhere except FrontPage) that can be linked to a page. For example, if a page lists a course syllabus, the titles for each week can be set as bookmarks and a row of links can be added to the top of the page that will each skip down to those bookmarked sections. This method of using bookmarks allows visitors to your site to quickly access information by not having to scroll down the page to view the information they want.

Add a bookmark to a page by following these steps highlighting the text or graphic that will be the bookmark and select **Insert|Bookmark...** from the menu bar. Enter the bookmark name in the space provided and click **OK**.



Create a link to a bookmark by highlighting the text that will be the link and pressing **CTRL+K**. Select the bookmark from the drop-down menu in the **Optional** category and click **OK**. Link to a bookmark on a separate page by first selecting the file name from the listing and then choosing from the bookmarks in the drop-down menu.



Hotspots

By making a graphic a link in the ways that have already been discussed, each graphic can only link to one location. However, you may have a single graphic that has several sections that each need to link to different pages. Hotspots allow you to do this by creating an image map over the graphic. The main header on the FGCU homepage will be used as an example:



The FGCU logo, "Search", "Directory", and "Index" images all link to different pages on this single graphic. Hotspots can be created by following these steps:

- Insert the graphic onto the webpage.
- Using the hotspot tools on the **Drawing** toolbar, use the necessary shapes to draw the hotspots on the graphic.



The rectangle tool will be used first to draw the hotspot around "search".



- The hyperlink window will appear when the mouse button is released. Enter the URL, e-mail address, or bookmark the hotspot will link to.
- Repeat steps 2-3 until all the hotspots have been added. Use the handles on the hotspots to resize them.



- If the remainder of the graphic (any part not covered by a hotspot such as the green to blue gradient in the center of this graphic) should be another link, right-click on any area of the graphic that is not a hotspot, select **Picture Properties** and enter the **Default hyperlink** location.

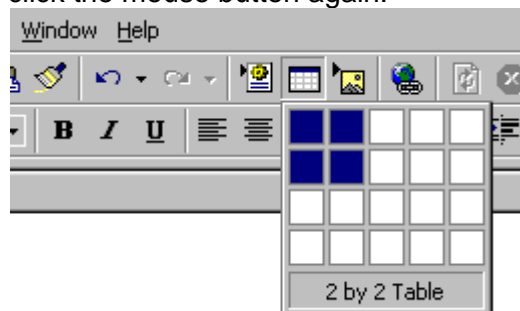
Use of Table

On web pages, tables can serve many functions:

- Page layout
- Displaying information in formatted tabular form
- Adding background color and borders to blocks of text

Creating a Table

A quick way to create a small table is using the table button on the standard toolbar. Click the button and drag the mouse over the grid, highlighting the cells that should appear on the table. When the table size has been selected, click the mouse button again.



A table outline with 2 rows and 2 columns will appear on the page:

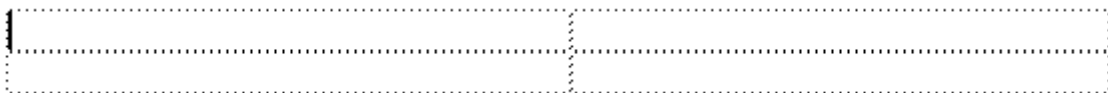


Table Properties

Select **Table|Properties|Table** from the menu border to modify the table's properties.



- **Alignment** refers to the table's position on the page, not the alignment of the text within the table. Choose "Center" to center the table on the page, or select left, right, or justify. Default is usually left alignment.
- **Cell padding** is the number of pixels between the text and the cell walls.
- **Cell spacing** is the number of pixels between the table cells.
- **Specify width** sets the width of the table by a distinct number of pixels or by a percentage of the screen width.
- **Specify height** is usually not necessary to set since the height depends on the number of rows in the table.

The following table was produced by the settings shown in the window above.

1	Microsoft FrontPage
2	Microsoft PowerPoint

- **Border size** indicates the depth of the table border. The dotted lines on the table above are shown only as a visual reference of the table structure, but since this table's border is set to 0 pixels, no borders will show on a web page:

1.	Microsoft FrontPage
2.	Microsoft PowerPoint

Below is the same table with a border set to 5 pixels:

1.	Microsoft FrontPage
2.	Microsoft PowerPoint

- **Border color** will change the color of the borders on the table. The MSIE and Netscape browsers read this property differently. MSIE changes the whole border to a solid color, while Netscape keeps the three-dimensional quality of the table and only changes the outer

border of the table. Since FrontPage is a Microsoft product, you will always see the MSIE version when constructing a web page in FrontPage.

Microsoft Internet Explorer 5.0

1.	Microsoft FrontPage
2.	Microsoft PowerPoint

Netscape Navigator 4.7

1.	Microsoft FrontPage
2.	Microsoft PowerPoint

- **Light border and dark border** change the highlight and shadow colors of the table. Be aware that these attributes are not read by Netscape. Light and dark borders of red and green have been added to the table, but notice that the Netscape table is still blue:

Microsoft Internet Explorer 5.0

1.	Microsoft FrontPage
2.	Microsoft PowerPoint

Netscape Navigator 4.7

1.	Microsoft FrontPage
2.	Microsoft PowerPoint

- **Background color** adds a background to the table cells. MSIE will add the color to the background of the cells and the space between the cells while Netscape only adds the color to the background of the cells:

Microsoft Internet Explorer 5.0

1.	Microsoft FrontPage
2.	Microsoft PowerPoint

Netscape Navigator 4.7

1.	Microsoft FrontPage
2.	Microsoft PowerPoint

- **Use background picture** will add a background image to the table and again, there are differences between browsers. MSIE will repeat the image over the entire background of the table while Netscape repeats the image in each cell:

Microsoft Internet Explorer 5.0

1.	Microsoft FrontPage
2.	Microsoft PowerPoint

Netscape Navigator 4.7

1.	Microsoft FrontPage
2.	Microsoft PowerPoint

- **Cell Properties**

Select **Tables|Properties|Cell** from the menu bar or **Cell Properties** from the shortcut menu to change the properties of the table cells. Begin by highlighting the cells whose properties will be changed.



- **Horizontal alignment** is defaulted to the left side of the table cell. Change this attribute to center or right-justify the text within the table cell.
- **Vertical alignment** is defaulted at middle as shown in the example below. Since the text in the right-hand column cover more than one line and the left-hand cells do not, that text is centered vertically in the cell. Select "top" or "bottom" to override this default setting.

1.	Microsoft FrontPage tutorial
2.	Microsoft PowerPoint tutorial

- Setting **rows spanned** and **columns spanned** is better achieved by a method explained below.
- **Specify width** and **specify height** will set the width and height of the cells. Percentages refer to the portion of the table, not a percentage of the entire screen.
- Select **Header cell** to automatically bold and center the content of the cell.
- **Border color** is a setting that is not read by Netscape. This changes the color of the cell border only when viewed with MSIE. Note the red borders on the cells in the top row of the MSIE example:

Microsoft Internet Explorer 5.0

1.	Microsoft FrontPage tutorial
2.	Microsoft PowerPoint tutorial

Netscape Navigator 4.7

1.	Microsoft FrontPage tutorial
2.	Microsoft PowerPoint tutorial

- Light and dark border settings will additionally be read by MSIE but not by Netscape.
- **Background color** changes the cell's background color. In Netscape, this is the same effect as setting the entire table's background color since the color is not added between the cells.

Microsoft Internet Explorer 5.0

1.	Microsoft FrontPage tutorial
2.	Microsoft PowerPoint tutorial

Netscape Navigator 4.7

1.	Microsoft FrontPage tutorial
2.	Microsoft PowerPoint tutorial

- **Background image** adds a graphic to the background of each cell. In Netscape, this is the same effect as setting the background graphic for the entire table since it begins a new repeating pattern for each cell.

Microsoft Internet Explorer 5.0

1.	Microsoft FrontPage tutorial
2.	Microsoft PowerPoint tutorial

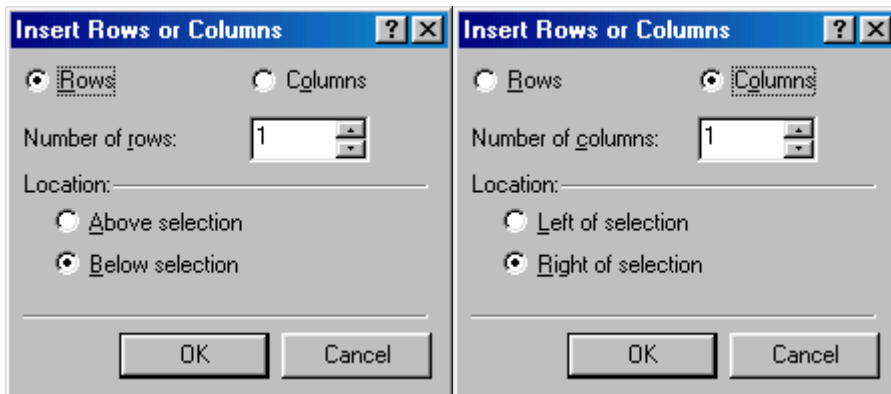
Netscape Navigator 4.7

1.	Microsoft FrontPage tutorial
2.	Microsoft PowerPoint tutorial

Inserting Rows and Columns

Quickly add rows or columns to a table by placing the cursor in the cell the new row or column will be adjacent to, right-click the mouse to access the popup shortcut menu, and select **Insert Row** or **Insert Column**. Another method is to use the **Insert Rows or Columns** dialog box:

- Place the cursor in a cell where the new row or column will be adjacent to.
- Select **Table|Insert|Rows or Columns** from the menu bar.



- To insert a row, select **Rows** and enter the **Number of Rows**. Then select the **Location** of the new row by selecting **Above selection** or **Below selection** from where you placed the cursor in step 1.
- Click **Columns** to insert a new column and the choices will change. Enter the **Number of columns** and the **Location** left or right of the selected point.
- Click **OK**.

Spanning Cells

There is often the need to create a cell that spans rows or columns, such as the title at the top of a table. This example will begin with the same table already used on this page.

- Insert a new row to the top of the table.
- Type the text of the row that will be spanned across the columns by typing into the first cell and highlight the cells as shown below:

FGCU Microsoft Office Tutorials	
1.	Microsoft FrontPage tutorial
2.	Microsoft PowerPoint tutorial

- Select **Table|Merge Cells** from the menu bar or right-click with the mouse and select **Merge Cells** from the popup shortcut menu..

FGCU Microsoft Office Tutorials	
1.	Microsoft FrontPage tutorial
2.	Microsoft PowerPoint tutorial

- Center the text in the cell from the **Cell Properties** window and deselect the cell.

FGCU Microsoft Office Tutorials	
1.	Microsoft FrontPage tutorial
2.	Microsoft PowerPoint tutorial

- To split the cell again, select the cell and choose **Table|Split Cell** from the menu bar.

Image Types

HTML code used on the web recognizes two basic graphic formats - GIF and JPEG. All the images on your web page must be either of these formats.

- **JPEG** (Joint Photographic Exchange Group) - As a general rule, photos should be saved as JPEGs. This file type consists of 16 million colors.
- **GIF** (Graphic Interchange Format) - These files contain 256 colors or less and should generally be used for non-photo graphics. All of the images on these tutorial pages are saved in GIF format.

Inserting a Graphic

To add a photo or graphic to a web page, select **Insert|Picture|From File** from the menu bar. Choose the file and click **OK**.

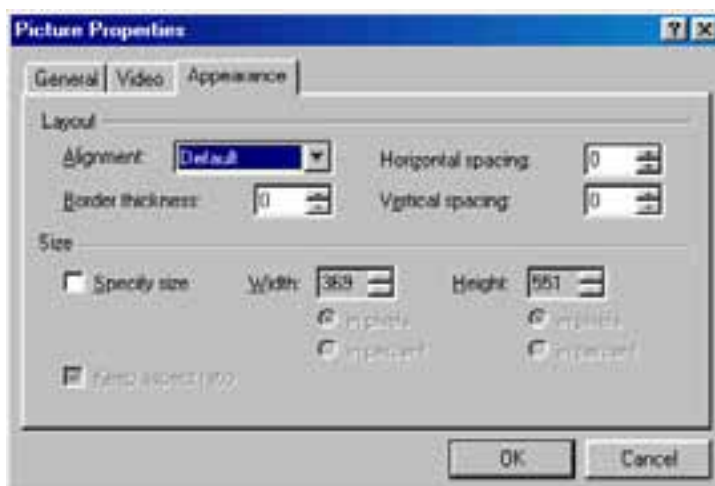
Picture Properties

To change the properties of the picture, select the image and choose **Format|Properties** from the menu bar.



Below **Alternative representations**, type a description of the image in the **Text** box. This text will appear in place of the graphic if the user browses your site with graphics turned off on their browser and will also be displayed while the picture is loading.

Click the **Appearance** tab.



- Change the **Alignment** if the picture should be aligned to the right or left of the text on the page or if it should be centered on the page.
- **Border thickness** will add a border to the picture depending on the number of pixels you enter. Leave this value at "0" if there is no border on the picture.

- **Horizontal spacing** and **Vertical spacing** are measured in pixels and will add white space surrounding the picture either above and below (vertical) or on both sides (horizontal).
- FrontPage automatically calculates the size of the image. However, if you would like it smaller or larger than actual size, check the **Specify size** box and enter the new **Width** and **Height** values. Please note, that it is recommended that you change the actual size of the image in an editing program such as Adobe Photoshop instead of changing these values, particularly if the actual size is large and you want it to appear smaller on the screen. Resizing the actual size of the graphic will lower the download speed of the image.

Horizontal Lines

Horizontal lines can divide sections of text for easy reading.

- Place the cursor on the page where the horizontal line should be added.
- Select **Insert|Horizontal Line** from the menu bar.
- Double click on the line to change its properties.



- Set the **Width** of the line either as a percentage of the window or an absolute size in pixels.
- Change the thickness of the line by setting the **Height** in pixels.
- Set the **Alignment** of the line to the left, center, or right size of the page.
- The lines are automatically shaded to give the illusion of depth. Check the **Solid line** box to make the line all one color and set a **Color** for the line if necessary (not supported by Netscape).

Shaded horizontal line:



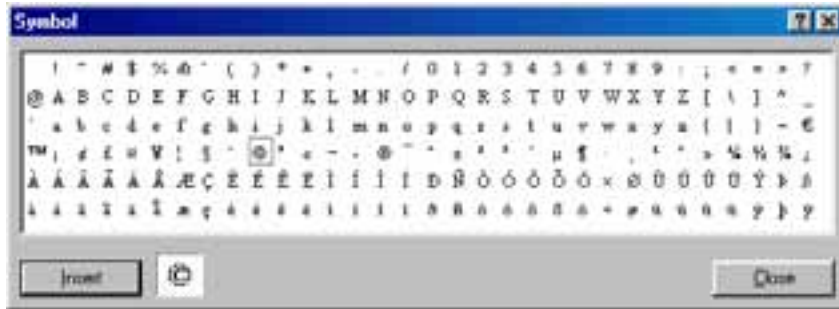
Solid horizontal line (red when viewed with MSIE):



- Click **OK** when finished.

Symbols

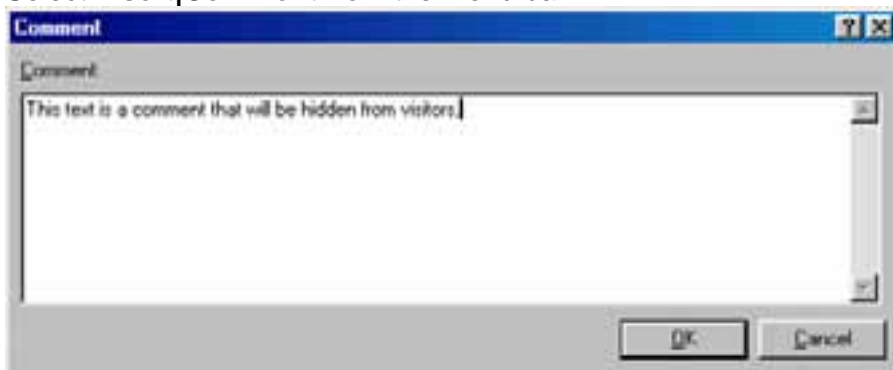
Add unique symbols such as the copyright (©) and accented letters (é) from the **Symbol** dialog box. Select **Insert|Symbol...** from the menu bar. Click the symbol on the list you want to add and it will appear in the preview window. Click the **Insert** button to add the symbol to the page. Keep adding symbols and click **Close** when finished.



Comments

Text can be hidden from visitors to your web page and still be visible to you when the page is edited in FrontPage. Follow these steps to add comments:

- Place the cursor on the page at the location where the comment will be added.
- Select **Insert|Comment** from the menu bar.



- Type the text in the **Comment** box and click **OK**.
- Notice that the commented text begins with "Comment: " and is another color to differentiate it from regular text. Open the page in a browser to see that the commented text is not visible on the page.

This text can be viewed by visitors to the web page.

Comment: This text is a comment that will be hidden from visitors.

Keyboard shortcuts can save time and the effort of constantly switching from the keyboard to the mouse to execute simple commands. Print this list of FrontPage keyboard shortcuts and keep it by your computer for quick reference.

Note: A plus sign indicates that the keys need to be pressed at the same time.

Action	Keystroke	Action	Keystroke
Document actions		Formatting	
Open a page	CTRL+O	Select all	CTRL+A
New page	CTRL+N	Copy	CTRL+C
Save	CTRL+S	Cut	CTRL+X
Print	CTRL+P	Paste	CTRL+V
Properties	ALT+ENTER	Undo	CTRL+Z
Refresh	F5	Redo	CTRL+Y
Spelling	F7	Bold	CTRL+B

Thesaurus	SHIFT+F7
Line break	SHIFT+ENTER
Remove formatting	CTRL+Spacebar

Editing	
Find	CTRL+F
Replace	CTRL+H
Insert hyperlink	CTRL+K
Spell checker	F7
Macros	ALT+F8

Italics	CTRL+I
Left justified	CTRL+L
Center justified	CTRL+E
Right justified	CTRL+R
Decrease indent	CTRL+SHIFT+M
Increase indent	CTRL+M

Web Design Tips

- **Background Images:** Use background images with caution. Light watermarks usually work fine, but dark, busy graphics can impair the readability of the page. Solid, muted colors are usually best to use.
- **Fonts:** Stick with common fonts such as Arial and Times New Roman. Although there are many fonts to choose from, if the user does not have a font to choose from on his or her computer, a default font will be used.
- **Long Pages:** Divide the information into different pages. A long and endless scrolling page is difficult to read.
- **Sound:** Refrain from using sound, particularly embedded sound files, if possible. If a sound file must be included, make it a link on the page so that the visitor to your page can turn it on and off. Many people surf the web with their computer speakers turned off so embedding a sound file that automatically downloads will unnecessarily increase the download time of the page.
- **Hyperlinks:** Always use descriptive words for link text instead of the simple "Click Here!" For example:

Incorrect - This site includes tutorials on the use of programs in the Microsoft Office suite.

To view the Microsoft Office 2000 tutorials, [click here](#).

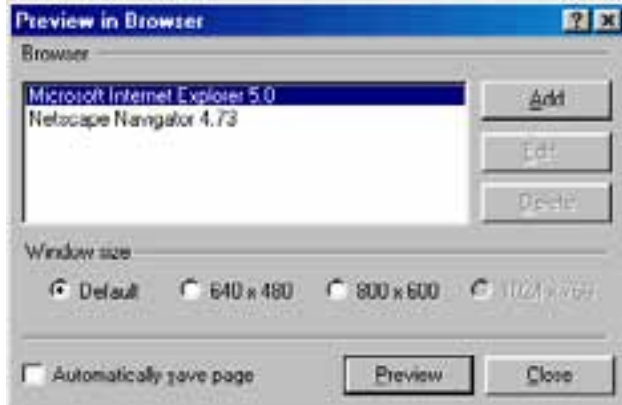
Correct - To learn more about the use of programs of the Microsoft Office suite, visit the [Microsoft Office 2000 tutorial homepage](#).

- **Page Size:** While most Internet users use a screen resolution of 800X600 or higher, there are still users with monitors set to 640X480. Design your web pages so that all material is viewable on this small resolution. Therefore, do not create graphics or tables wider than 620 pixels.

Testing

Since different browsers (mainly Microsoft Internet Explorer and Netscape Navigator) read webpage differently, preview your pages in both browsers to make sure the page looks the way you want it. FrontPage provides an easy way to accomplish this.

- Select **File|Preview** in Browser from the menu bar.



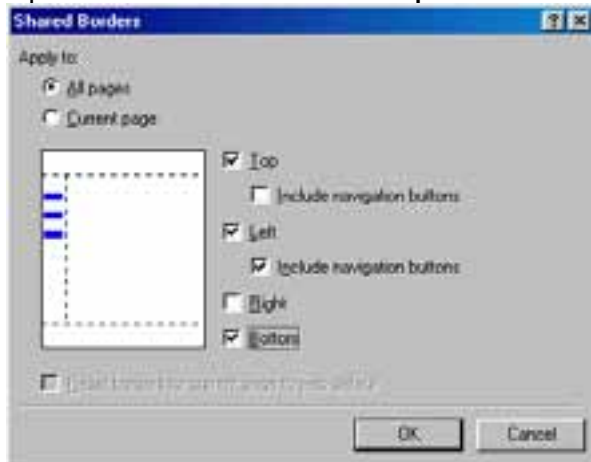
- The list of browsers installed on your computer will be listed. Highlight the browser name and click **Preview** for each browser.

Shared Borders are parts of the web page that share content with the other pages in the web. They are located at the top, bottom, left side, or right side of the page and are useful for information that should appear on every page in the web such as navigation links, web site headers, and copyright information.

Adding Shared Borders

Add shared borders to a web by following these steps:

- Open a web and select **Format|Shared Borders...** from the menu bar.



- Check **All pages** to apply the border(s) to every page in the web or **Current page** if the border(s) should appear only on the current page.
- Select the areas where the shared borders should appear by checking the Top, Left, Right, and Bottom boxes. It is not recommended to have content for the Right border since it may not be visible to users with low screen resolutions.
- Check the **Include navigation buttons** boxes if those sections will have site navigation elements (explained in detail on the next page).
- Click **OK** when finished. These properties can be changed by selecting **Format|Shared Borders...** again from the menu bar after the shared borders have been defined.

Shared Border Content

After completing the steps outlined above, a setup similar to the image below will appear on the web page:

Comment: This border appears in all pages in your Web....

[Add this page to the Navigation view to display hyperlinks here]

Comment: This border appears in all pages in your web....

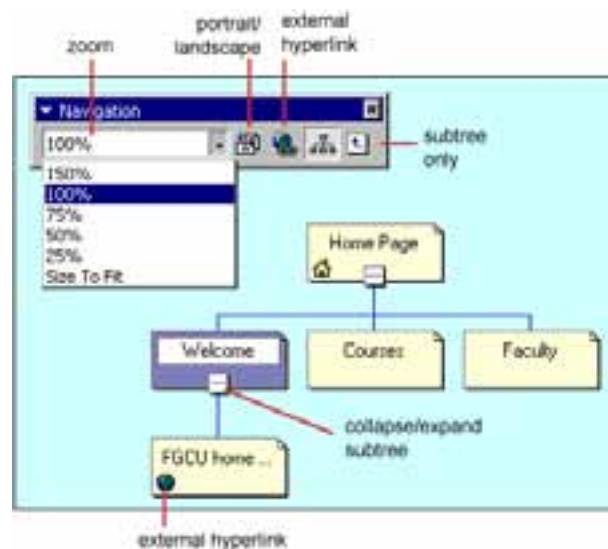
Edit content by clicking on the shared border and replacing purple comment text with text and graphics.

Welcome to FGCU

[Edit the properties for this Navigation Bar to display hyperlinks here]

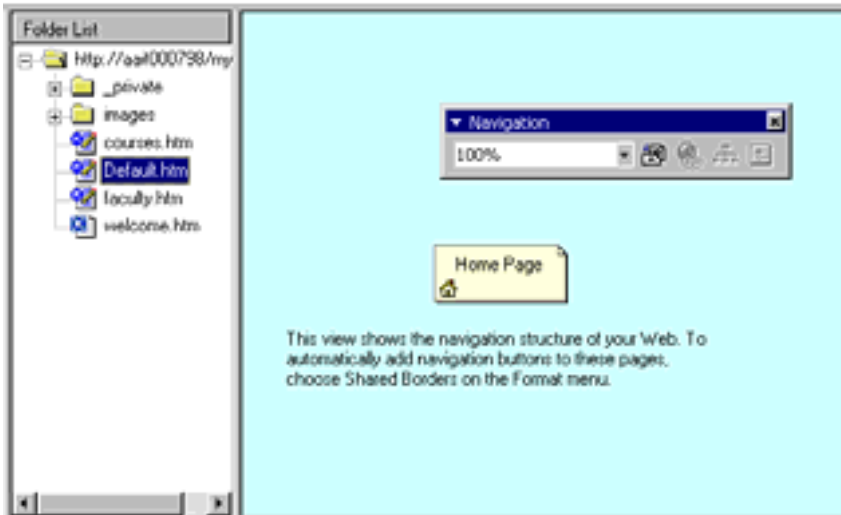
Navigation bars are automatically created by FrontPage according to the navigation structure that is set up in Navigation View. These are explained in detail on the next page...

Navigation bars created in FrontPage are a quick method for adding navigation links to all the pages in a web. The diagram below illustrates the features of creating navigation bars:

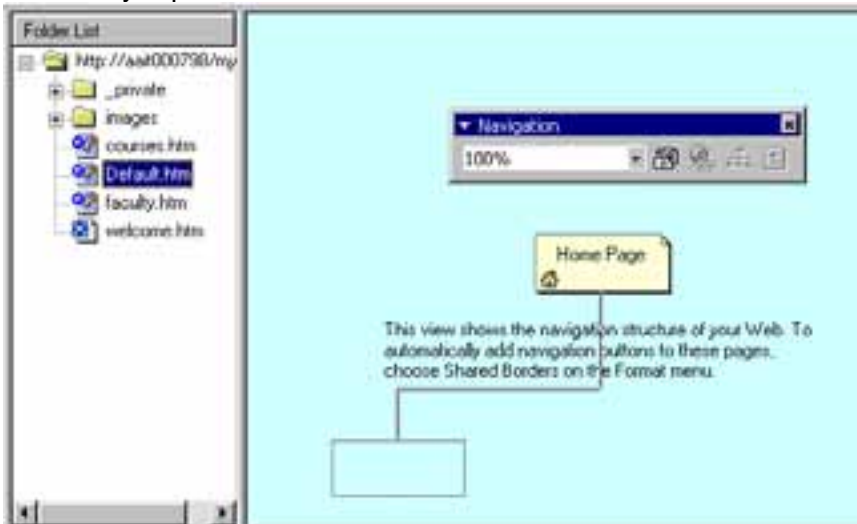


Create a Navigation Bar

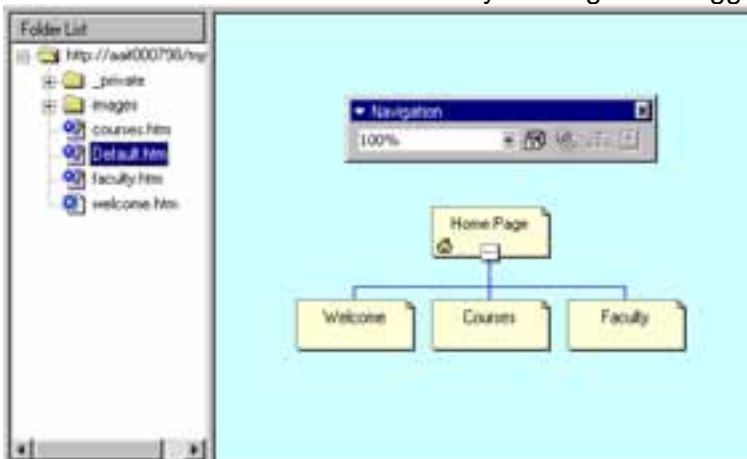
- Open a web in **Navigation View**. A single "parent" page should be visible in the blue area of the screen.



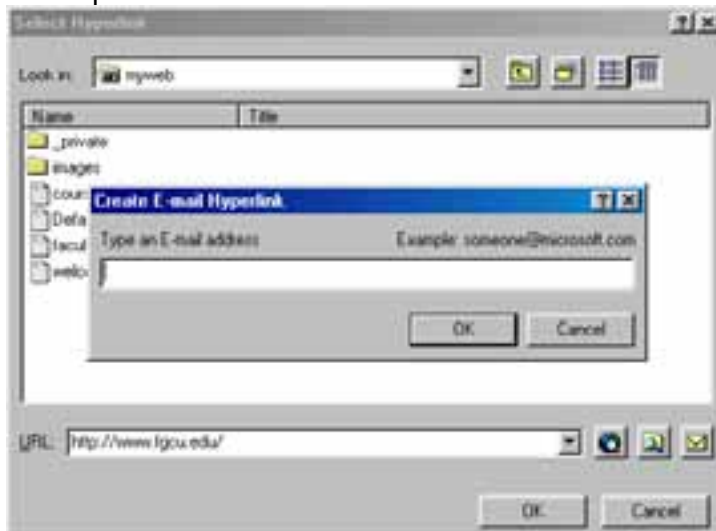
- Add more pages to the navigation tree by dragging the file names in the **Folder List** onto the blue navigation window. Relation connectors will be added between the new "child" page and the parent page. Release the mouse button when the relationship between the pages is accurately represented.



- Continue adding pages to the navigation tree. If you make a mistake, pages can always be moved to new locations on the tree by clicking and dragging them with the mouse.

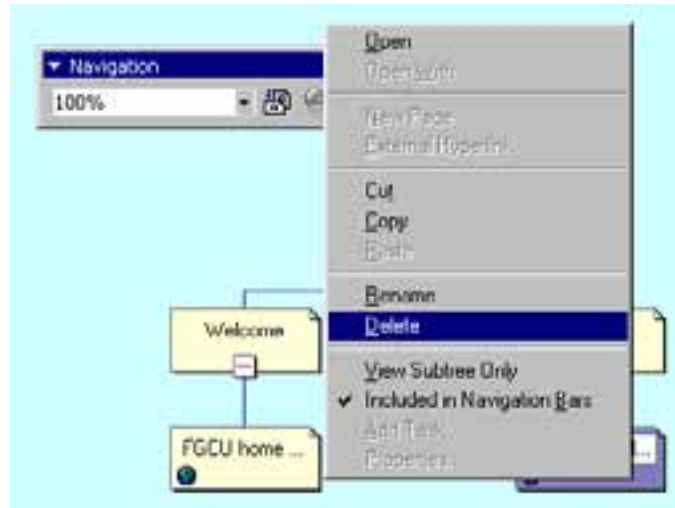


- Add a link to a page outside of the web by selecting the page the link will extend from on the navigation tree and clicking the external hyperlink button on the Navigation toolbar. Type the URL beginning with "http://" in the **URL:** window of the **Select Hyperlink** box. If the link is a e-mail address, click the e-mail button at the end of the URL line that shows the image of an envelope. Enter the e-mail address in the **Create E-mail Hyperlink** box and click **OK**.

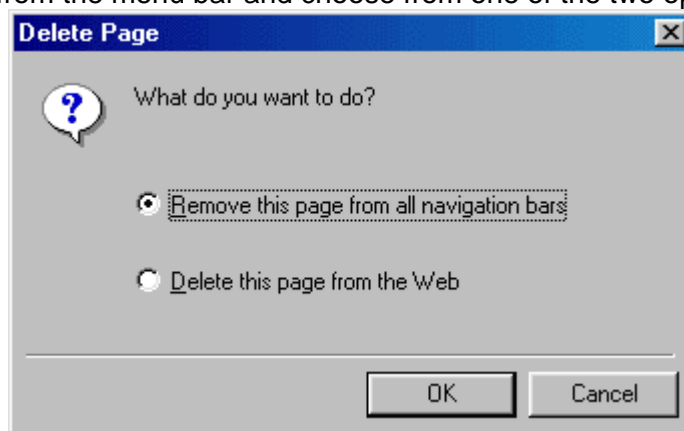


Deleting Pages from the Navigation Tree

Delete a page from the navigation tree by right-clicking on the page icon with the mouse and selecting **Delete** from the popup shortcut menu. To keep a page on the navigation tree but have it not appear on the navigation bar, select **Included in Navigation Bars**. The page now appears gray on the diagram.



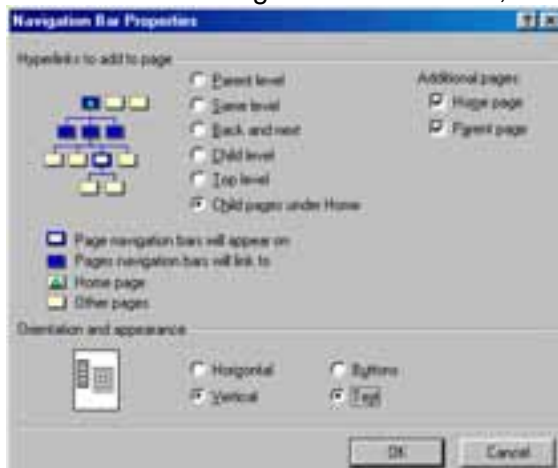
Or, select **Edit|Delete** from the menu bar and choose from one of the two options:



View portions of the tree by clicking the **Collapse/Expand Subtree** buttons between pages. These buttons will show a minus sign (-) when the subtree can be collapsed and a plus sign (+) when the subtree is already collapsed and can be expanded. View only the subtree of a page by highlighting the page and clicking the **Subtree Only** button on the Navigation toolbar. The entire tree structure can also be toggled between landscape and portrait by clicking the orientation button on the Navigation toolbar.

Navigation Bar Properties

- Open a web page with a navigation bar in **Page View**.
- Right click on the area that the navigation bar appears or, if the navigation bar has not yet been created through Shared Borders, select **Insert|Navigation Bar...** from the menu bar.



- Select which pages should appear as links in the navigation bar by selecting an option from **Hyperlinks to add to page**. The tree image will give you a preview.

- Check the **Home page** and **Parent page** boxes if those **Additional pages** should be added to the navigation bar.
- Select the **Orientation and appearance** by choosing a **Horizontal** or **Vertical** layout for the links and if they should appear as **Buttons** or **Text**.
- Click **OK** when finished.

Cascading Style Sheets (CSS) allow you to format a web page by setting font attributes such as small caps and changing the character spacing, paragraph properties, and borders and shading for text boxes. Style sheets can be applied to a web page in three ways:

- **Embed a style sheet** on a web page by listing the style attributes at the top of the page.
- Apply **inline styles** within the web page to add a style to individual elements of a page.
- Link to an **external style sheet** if several pages will have the same styles. By using this method, style attributes are only changed in one location and the changes are reflected on many pages. These external style sheets are saved as a separate file with the file extension .css.

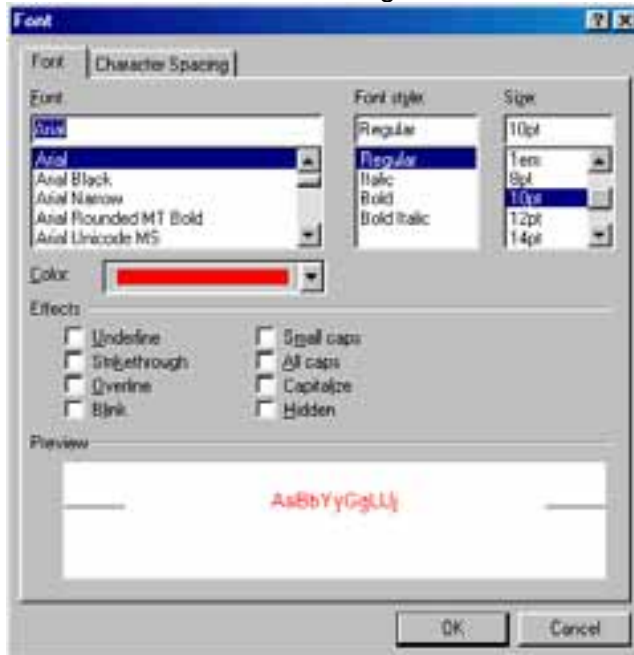
Create an Embedded Style

- Select **Format|Style** from the menu bar.
- Click the **New** button to create a new style.
- Type a period (.) followed by a name containing no spaces for the new style in the **Name (selector)** field. Below is an example of a style that will create red text.



- Click the **Format** button to select an element the style formatting will apply to. In this example, the font color will be changed first so **Font...** is selected from the menu.

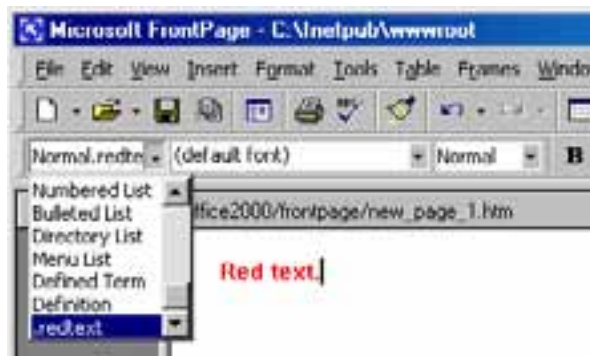
- From the **Font** dialog box, the font has been changed to Arial, size to 10 point, and color to red. Click **OK** when the changes have been made.



- Select other options from the **Format** button menu to change more element attributes. When all the styles have been selected, click **OK** on the New Style window and **OK** on the Style window.

Apply the New Style

To apply the style you have just created, highlight the text that the style will be applied to. The style is listed in the style menu on the formatting toolbar. Click the window and scroll down to select the style.



Inline Styles

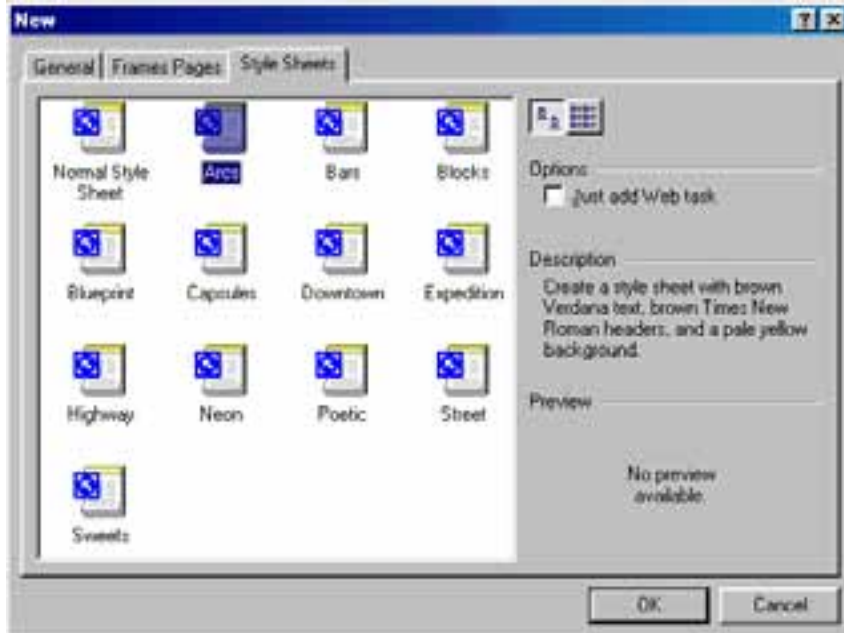
Some formatting styles, such as font properties, borders and shading, are automatically applied to certain elements such as portions of text, paragraphs, and divisions. Other styles that are applied to tables and form elements can be manually added as described below:

- Highlight the table or form element and select **Format|Properties** from the menu bar.
- Click the **Style...** button on the properties window.
- Styles you have already created are listed in the **Class** drop-down menu. Select one of these styles or click the Format button to change another property.
- Press **OK** when finished.

Using Preset External Style Sheets

FrontPage comes with several preset styles that can be added to web pages. It is also helpful to review the code of these preset style sheets to gain a better understanding of CSS when you create your own style sheets. Follow the steps outlined below to save one of these styles as an external CSS to use on a web page.

- Select **File|New|Pages** from the menu bar and click the **Style Sheets** tab.

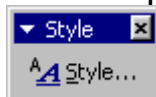


- Highlight the choices to view descriptions for each of the styles and click **OK**.
- The style sheet code will appear in the window. Save the style sheet using the .css extension.

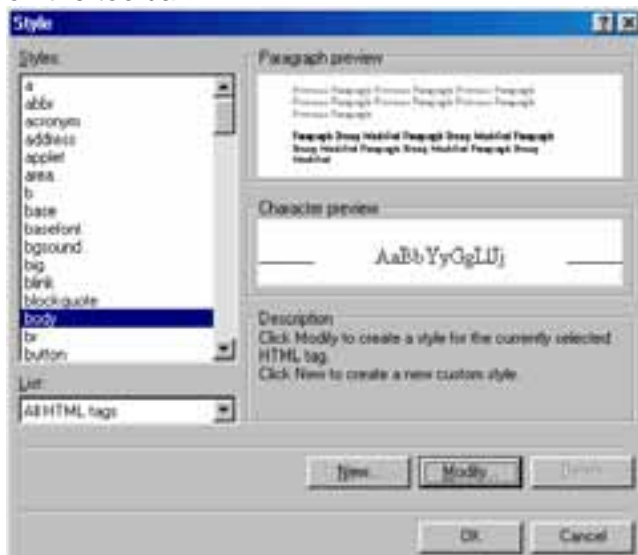
Create an External Style Sheet

To create an external style sheet from scratch, follow these steps:

- Select **File|New|Pages**, click the **Style Sheets** tab, and choose **Normal Style Sheet**.



- A blank page will appear on the screen with a small **Style** toolbar. Click the **Style...** button on the toolbar.



- Highlight an element from the **Styles** list that the style will be added to and click the **Modify...** button. For example, select the "body" tag from the list and we will change the default text style for the page.
- Click the **Format** button on the **Modify Style** window and select the elements that will be formatted. To change the default text style in this example, select "Font...". Choose "Arial" from the font list and "10pt" from the size listing.

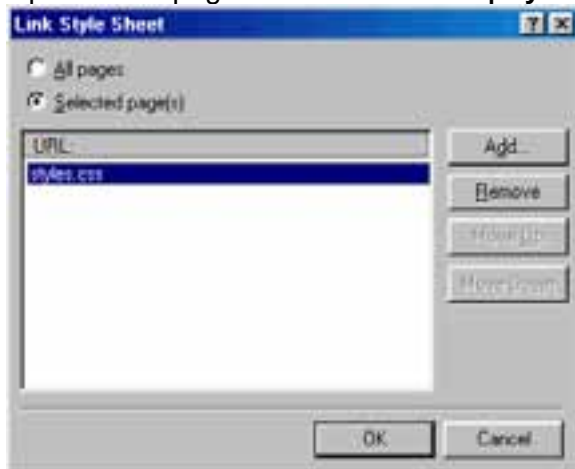
TIP - As discussed in the [text tutorial](#), it is advantageous to include several similar fonts in case a visitor to your site does not have the particular font you chose on their computer or is using a different operating system. Similar sans-serif fonts are Arial, Verdana, Geneva, and sans-serif while serif fonts are Times New Roman and Times. Type the font names in the **Font** box above the scrolling menu, separating each name with a comma.

- Click **OK** when finished.
- Click **OK** on the **Modify Style** window.
- Notice on the style window that this style is now listed. Click **OK** to exit the **Style** window or select another element to modify.
- Save the style sheet in .css format.

Link to an External Style Sheet

After you have created an external style sheet, it must be linked to a web page for the styles to be applied to the page.

- Open a web page and select **Format|Style Sheet Links** from the menu bar.



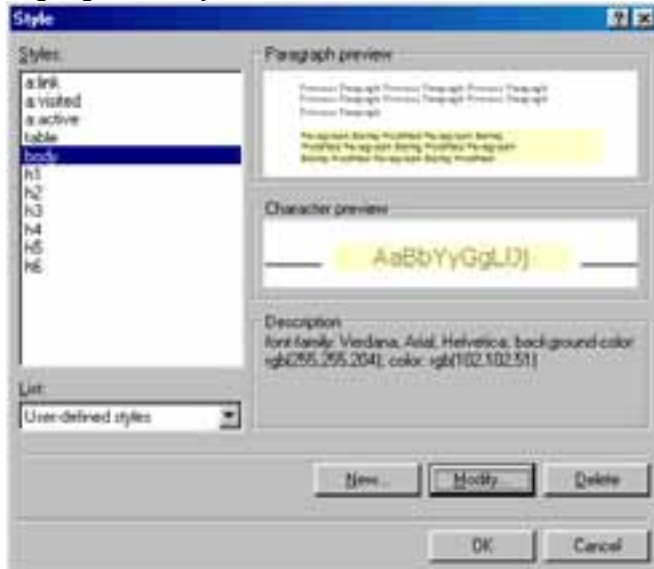
- Click the **Add...** button on the **Link Style Sheet** dialog box.
- Select the .css file and click **OK**.

To remove an external style sheet link from a page, highlight the style sheet in the list and click the **Remove** button.

Edit an External Style Sheet

- Open the .css file in FrontPage.
- Select **Format|Style** from the menu bar.

- Highlight the style from the list that needs to be changed and click the **Modify** button.



- Make the necessary changes by selecting options from the **Format** button menu.
- Click **OK** when finished and save the style sheet.

The themes provided in FrontPage can be modified to fit your needs. If you have already assigned a theme to a page that you would like to modify, follow the steps on this page. First select **Format|Themes** and click the modify button. This action will reveal additional buttons for modifying themes.



Colors

Click the **Colors...** button to modify the color scheme.

- **Color Schemes tab** - Select preset color schemes from the list and preview those colors in the theme in the Sample of Theme window.
- **Color Wheel tab** - Click and drag the circle in the color wheel to alter the color scheme. Use the Brightness slider to brighten and dim the colors.
- **Custom tab** - Change each text type individually by selecting the item from the drop-down menu and assigning a color.

- Click **OK** when you are finished modifying the color scheme.



Graphics

The bullets, banners, and backgrounds on a theme can be changed as well. Click the **Graphics...** button to change the images in the theme.

- Select items from the drop-down menu. Change the images from the text field provided under the **Picture tab** and modify the fonts used from the **Font tab**.
- At the bottom of the window, select "Normal Graphics" for static images and "Active Graphics" for Java rollover applets.
- Click **OK** when finished.



Text

The **Text...** button will provide a menu that will allow you to change the font of elements in the theme.

- Select a text element from the **Item** drop-down menu and then select the new font for the item from the **Font** list.

- Click the **More Text Styles...** button to change additional text properties.



Saving a Modified Theme

If you plan on using the same modified theme on other pages, the theme can be saved. Click the **Save As** button on the **Themes** dialog box and enter a name for the theme.

What Are Frames?

Frames divide a web page into sections where each has a different HTML source page and its own set of scroll bars. They can be useful for any site that requires part of the screen to remain static while the remainder of the screen can be scrolled. One example is site navigation where links can be placed in one frame and the scrolling page content is placed in another. There are several disadvantages in the use of frames including slower download time and problems with linking and printing so be sure to use them only if necessary.

For example, the Web Boards used for many courses at FGCU use frames. The black navigation bar, Conferences list, and main content frame are the three independent frames, each having their own HTML source page.

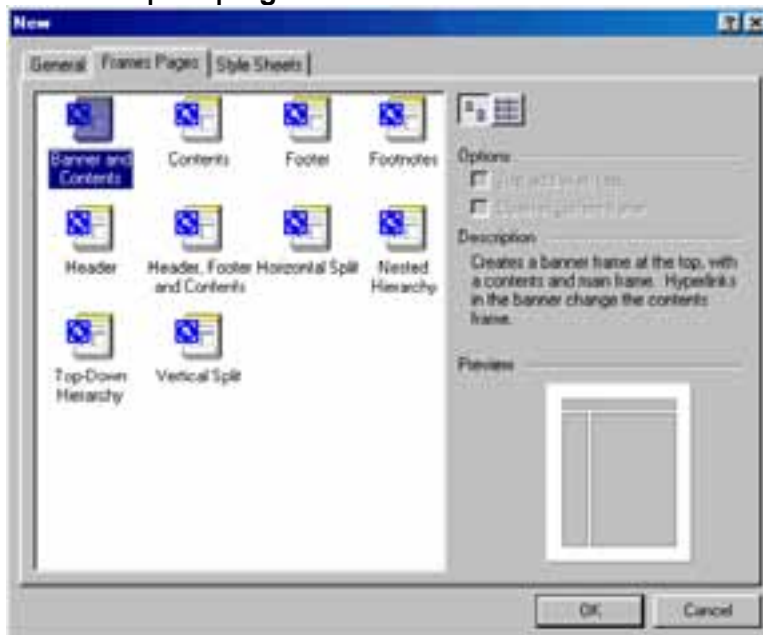


Frame pages actually consist of several HTML pages and the exact number depends on the number of frames on the page. The main page is called the *parent page*. This web page contains the instructions for the format and location of the frames, and a link to the initial loading page for each frame, but does not include the actual text or graphics that appear on the page. Each frame then has its own HTML source page that contains the text and graphics for that frame. Therefore, the WebBoard page displayed above actually consists of four pages: the parent and three frame source pages. You will need to keep this structure in mind when you save a frames page as FrontPage will prompt you to save each of these pages.

Create a Frames Page

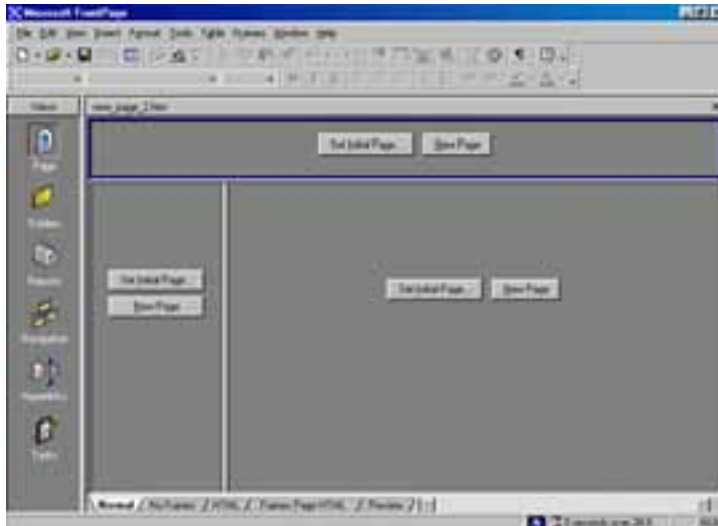
Create a frames page by following these steps:

- Select **File|New|Page** from the menu bar and click the **Frames Pages** tab.

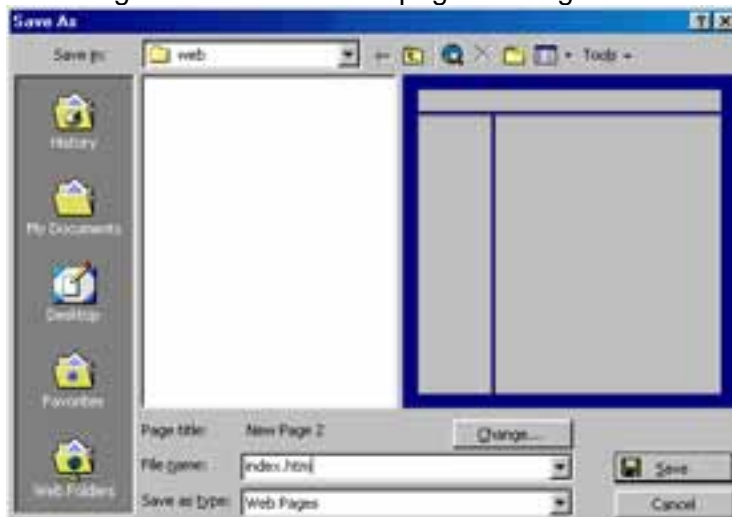


- Preview each of the choices by clicking on an icon once with the mouse and seeing the **Preview** window. Select the icon of the frames format you would like to use and click **OK**.
- When viewed in **Page** view, the web page will be divided into frames and each frame will have "Set Initial Page..." and "New Page" buttons. Click **New Page** if the source page for the frame does not yet exist. The frame will immediately turn white after the button is clicked and you will be able to type and add graphics just like a normal web page. Click **Set Initial Page** if the source page for the frame has already been created and select the file from the dialog

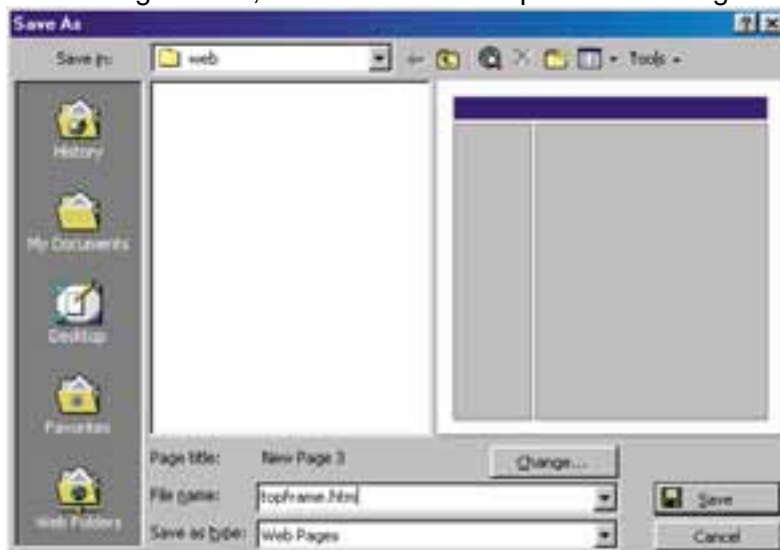
box.



- Save the frames pages by selecting **Files|Save As** from the menu bar. You will be prompted to save the main frame page first followed by each of the frame source pages. The diagram will highlight the page you are saving. Below, the diagram highlights all of the frames in blue, indicating that the main frame page is being saved:

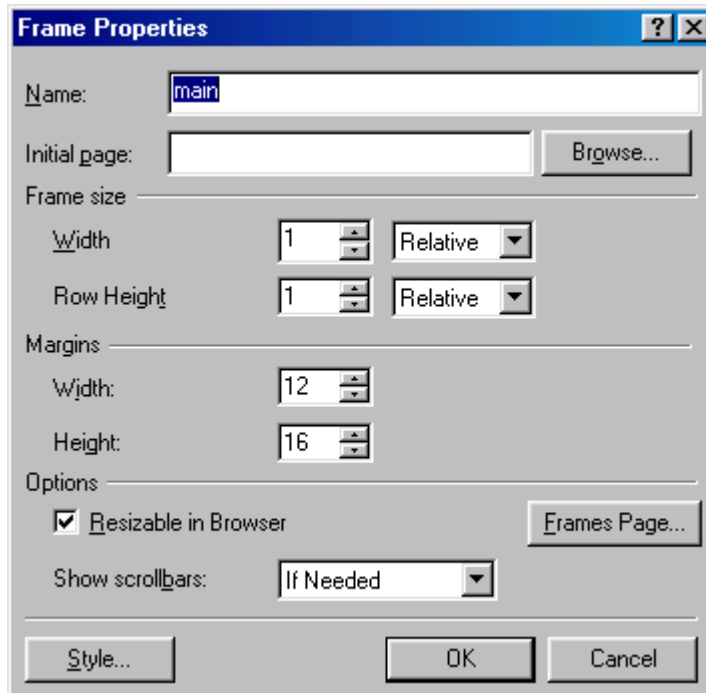


In the image below, the source of the top frame is being saved:



Frame Properties

Right-click on a frame either before or after its content page has been identified and select **Frame Properties** from the shortcut menu.



Name

Assign a name to each frame for linking purposes.

Initial Page

Assign the initial HTML source page.

Frame size

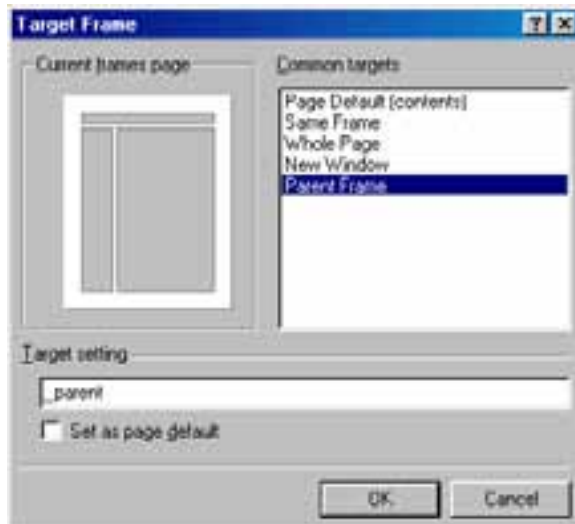
Designate the width and height of the frame in absolute pixels or as a percentage of the screen.

Options

Check "Resizable in Browser" if the user should be able to click and drag the frame borders to resize them. Make a selection from "Show scroll bars" if scroll bars should be visible in the frame.

Linking in Frames

When creating a link from a frames page, click the pencil button next to the **Target frame** option on the **Create Hyperlink** dialog box. Select the proper target for the link from the **Common targets** box.



Page Default will load the page in the default frame indicated in parentheses.

Same Frame will load the new page in the same frame.

New Window will open a new browser window.

Parent Window will load the page in the current window.

No Frames Page

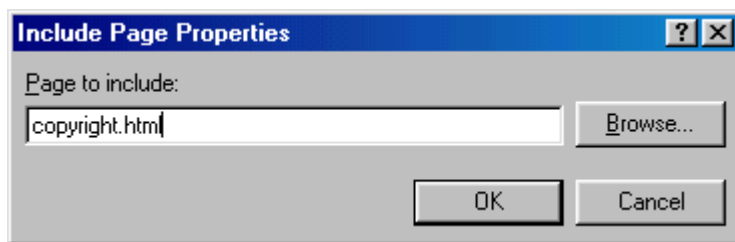
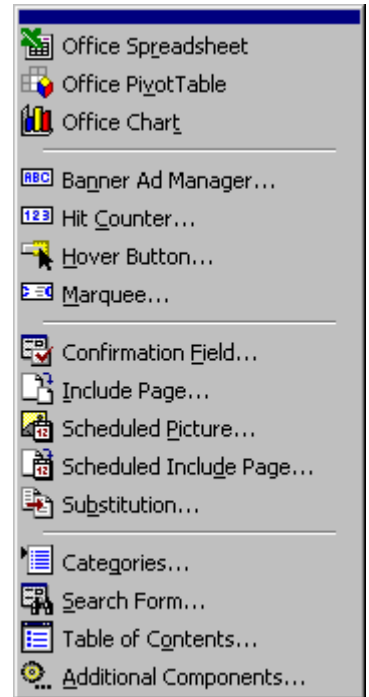
Early versions of browsers do not support, so it is necessary to prepare a page for visitors using these browsers. Build the "No Frames" page from the tab at the bottom of the screen. Use this page to link to individual main frame source pages in your site or provide links to download sites for the latest versions of Netscape Navigator and Microsoft Internet Explorer.

Access the Components menu pictured to the right by selecting **Insert|Component** from the menu bar. To make the menu its own floating toolbar, click and drag the blue bar at the top of the menu. You must save the web page first to be able to use all of the components listed.

Includes

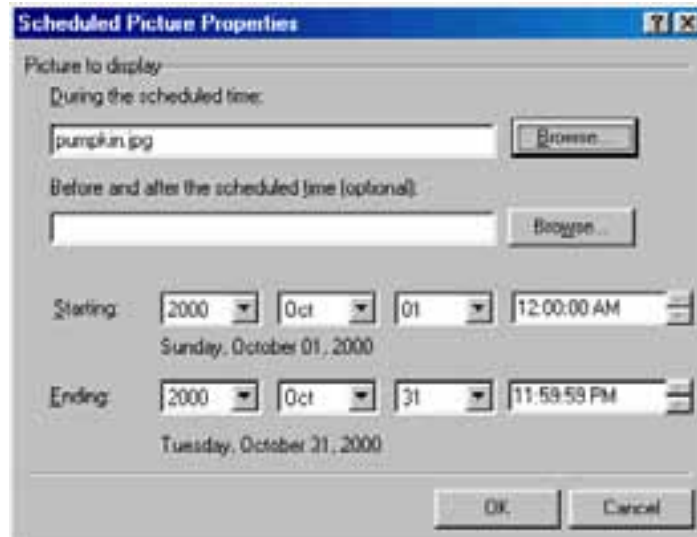
The Include feature allows you to display one page within another. This can often be helpful when placing copyright notices or menu links that appear identically on many pages. The copyright notice, for example, can be typed into a separate file and then that one file can be linked to all the pages in the web site. When the copyright changes, only the include file needs to be changed instead of correcting the copyright text on every individual web page.

To add an include page to another web page, first place the cursor on the page where the include file's contents should appear. Then, select **Insert|Component|Include Page** from the menu bar. Click the **Browse...** button to select the page you want to include and click **OK**. The include page's contents will appear on the destination page, but the contents can only be modified by opening the include page separately.



Scheduled Picture

Setting scheduled elements will automatically change page content on a given date. Add a scheduled picture by selecting **Insert|Component|Scheduled Picture...** from the menu bar. Select the picture and choose an optional image to show before and after the scheduled time. Then, set the starting and ending time period. In this example, a picture of a pumpkin is added to the site for the entire month of October. Click **OK** when finished.

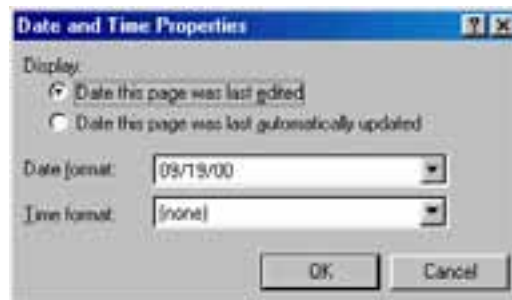


Scheduled Include Page

A scheduled Include page can be added in a similar way that a scheduled picture is inserted. Select **Insert|Component|Scheduled Include Page...** from the menu bar, select the file, and set the time period.

Date and Time

It is always advantageous to include a "last updated" date on the bottom of the web page so that visitors know how recent the material is. FrontPage can automatically update this date whenever the page is saved. Place the cursor on the page where the date should appear and select **Insert|Date and Time...** from the menu bar. If the page is set up to automatically update and this date should be reflected as the "last update", check the "Date this page was last automatically updated" box. Select a **Date format** and if the time should also appear, select the **Time format**. Click **OK** when finished.

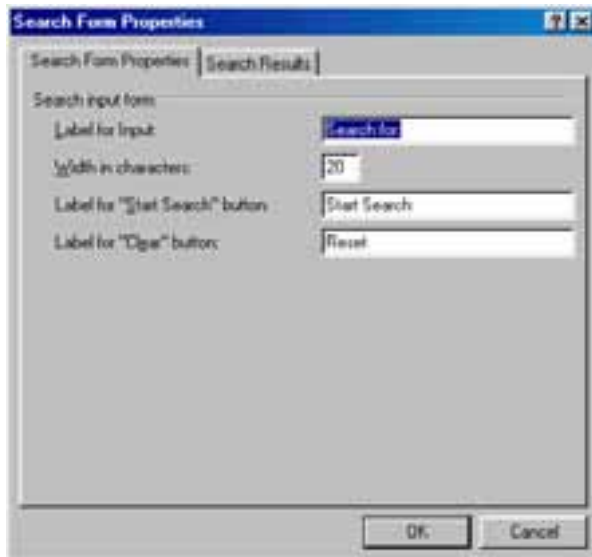


Search Form

Add a search form to the web site by selecting **Insert|Component|Search Form** from the menu bar. This feature will automatically create a simple search form:



Customize the search form from the properties window.

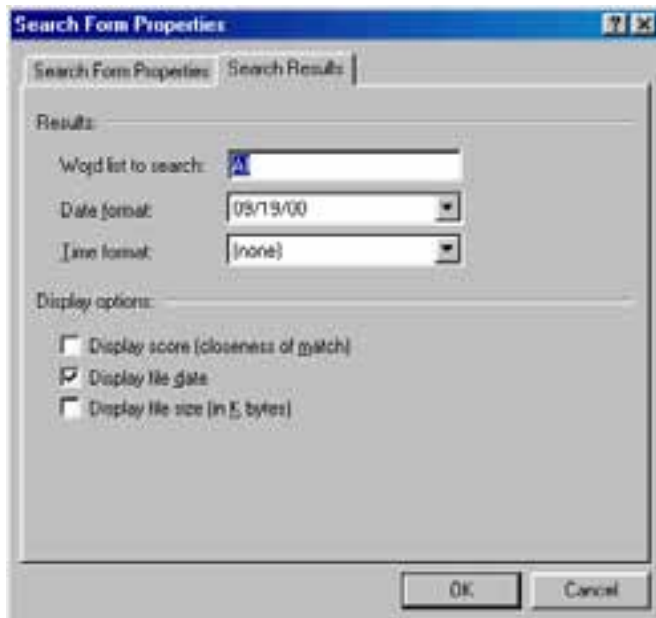


Label for Input is the text that appears before the search text box

Width in characters is the width of the search text box

Labels for "Start Search" and "Clear" buttons the text that appear on the buttons.

Click the **Search Results** tab to format the search results page.



Select the date and time format for displaying the results.

Display score is the closeness of the keywords entered to the page that was found. A higher score indicates a closer match.

Display file date includes the date the page was last modified

Display file size prints the size of the page in kilobytes

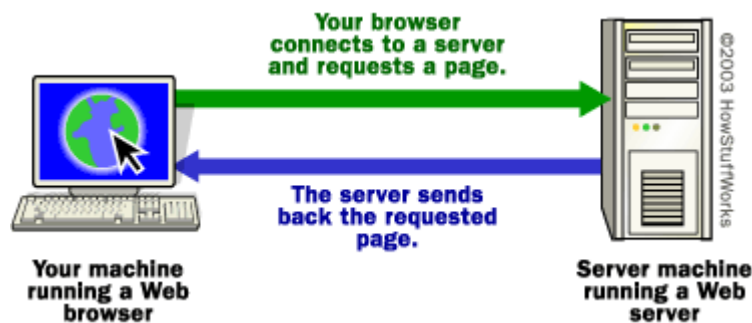
Requirements of Web Publishing ISP

ISP stands for Internet Service Provider. This is a company that provides an Internet connection. Small ISPs provide service via modem and ISDN while the larger ones also offer private line hookups (T1, fractional T1, etc.). The major online services such as America Online and CompuServe provide Internet access but are still known as "online services", not ISPs. They offer the members only content, forums and services in addition to Internet access.

Web Server

Let's say that you are sitting at your computer, surfing the Web, and you get a call from a friend who says, "I just read a great article! Type in this URL and check it out. It's at <http://computer.howstuffworks.com/web-server.htm>." So you type that URL into your browser and press return. And magically, no matter where in the world that URL lives, the page pops up on your screen.

At the most basic level possible, the following diagram shows the steps that brought that page to your screen:



Your browser formed a connection to a Web server, requested a page and received it.

Behind the Scenes

If you want to get into a bit more detail on the process of getting a Web page onto your computer screen, here are the basic steps that occurred behind the scenes:

The browser broke the URL into three parts:

The protocol ("http")

The server name ("www.howstuffworks.com")

The file name ("web-server.htm")

The browser communicated with a name server to translate the server name "www.howstuffworks.com" into an IP Address, which it uses to connect to the server machine.

The browser then formed a connection to the server at that IP address on port 80. (We'll discuss ports later in this article.)

Following the HTTP protocol, the browser sent a GET request to the server, asking for the file "http://computer.howstuffworks.com/web-server.htm." (Note that cookies may be sent from browser to server with the GET request -- see How Internet Cookies Work for details.)

The server then sends the HTML text for the Web page to the browser. (Cookies may also be sent from server to browser in the header for the page.)

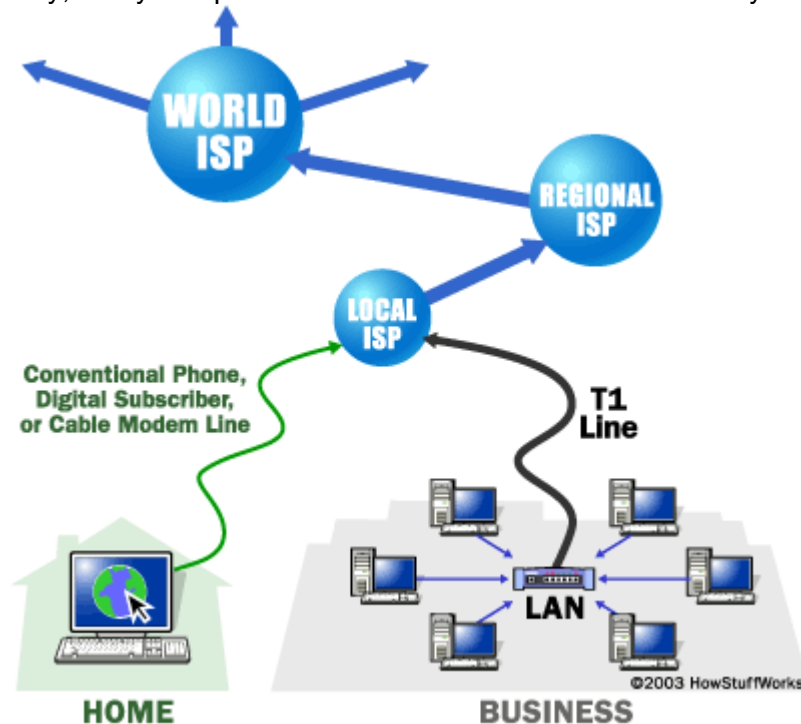
The browser reads the HTML tags and formats the page onto your screen.

If you've never explored this process before, that's a lot of new vocabulary. To understand this whole process in detail, you need to learn about IP addresses, ports, protocols... The following sections will lead you through a complete explanation.

The Internet

So what is "the Internet"? The Internet is a gigantic collection of millions of computers, all linked together on a computer network. The network allows all of the computers to communicate with one another. A home computer may be linked to the Internet using a phone-line modem, DSL or cable modem that talks to an Internet service provider (ISP). A computer in a business or university will usually have a network interface card (NIC) that directly connects it to a local area network (LAN) inside the business. The business can then connect its LAN to an ISP using a high-speed phone

line like a T1 line. A T1 line can handle approximately 1.5 million bits per second, while a normal phone line using a modem can typically handle 30,000 to 50,000 bits per second. ISPs then connect to larger ISPs, and the largest ISPs maintain fiber-optic "backbones" for an entire nation or region. Backbones around the world are connected through fiber-optic lines, undersea cables or satellite links (see An Atlas of Cyberspaces for some interesting backbone maps). In this way, every computer on the Internet is connected to every other computer on the Internet.



IP Addresses

To keep all of these machines straight, each machine on the Internet is assigned a unique address called an IP address. IP stands for Internet protocol, and these addresses are 32-bit numbers, normally expressed as four "octets" in a "dotted decimal number." A typical IP address looks like this:

216.27.61.137

The four numbers in an IP address are called octets because they can have values between 0 and 255, which is 28 possibilities per octet.

Every machine on the Internet has a unique IP address. A server has a static IP address that does not change very often. A home machine that is dialing up through a modem often has an IP address that is assigned by the ISP when the machine dials in. That IP address is unique for that session -- it may be different the next time the machine dials in. This way, an ISP only needs one IP address for each modem it supports, rather than for each customer.

If you are working on a Windows machine, you can view a lot of the Internet information for your machine, including your current IP address and hostname, with the command WINIPCFG.EXE (IPCONFIG.EXE for Windows 2000/XP). On a UNIX machine, type nslookup at the command prompt, along with a machine name, like www.howstuffworks.com -- e.g. "nslookup www.howstuffworks.com" -- to display the IP address of the machine, and you can use the command hostname to learn the name of your machine. (For more information on IP addresses, see IANA.)

As far as the Internet's machines are concerned, an IP address is all you need to talk to a server. For example, in your browser, you can type the URL <http://209.116.69.66> and arrive at the machine

that contains the Web server for HowStuffWorks. On some servers, the IP address alone is not sufficient, but on most large servers it is -- keep reading for details.

Domain Names

Because most people have trouble remembering the strings of numbers that make up IP addresses, and because IP addresses sometimes need to change, all servers on the Internet also have human-readable names, called domain names. For example, www.howstuffworks.com is a permanent, human-readable name. It is easier for most of us to remember www.howstuffworks.com than it is to remember 209.116.69.66.

The name www.howstuffworks.com actually has three parts:

The host name ("www")

The domain name ("howstuffworks")

The top-level domain name ("com")

Domain names within the ".com" domain are managed by the registrar called VeriSign. VeriSign also manages ".net" domain names. Other registrars (like RegistryPro, NeuLevel and Public Interest Registry) manage the other domains (like .pro, .biz and .org). VeriSign creates the top-level domain names and guarantees that all names within a top-level domain are unique. VeriSign also maintains contact information for each site and runs the "whois" database. The host name is created by the company hosting the domain. "www" is a very common host name, but many places now either omit it or replace it with a different host name that indicates a specific area of the site. For example, in encarta.msn.com, the domain name for Microsoft's Encarta encyclopedia, "encarta" is designated as the host name instead of www.

Name Servers

The whois Command

On a UNIX machine, you can use the whois command to look up information about a domain name. You can do the same thing using the whois form at VeriSign. If you type in a domain name, like "howstuffworks.com," it will return to you the registration information for that domain, including its IP address.

A set of servers called domain name servers (DNS) maps the human-readable names to the IP addresses. These servers are simple databases that map names to IP addresses, and they are distributed all over the Internet. Most individual companies, ISPs and universities maintain small name servers to map host names to IP addresses. There are also central name servers that use data supplied by VeriSign to map domain names to IP addresses.

If you type the URL "<http://computer.howstuffworks.com/web-server.htm>" into your browser, your browser extracts the name "www.howstuffworks.com," passes it to a domain name server, and the domain name server returns the correct IP address for www.howstuffworks.com. A number of name servers may be involved to get the right IP address. For example, in the case of www.howstuffworks.com, the name server for the "com" top-level domain will know the IP address for the name server that knows host names, and a separate query to that name server, operated by the HowStuffWorks ISP, may deliver the actual IP address for the HowStuffWorks server machine.

On a UNIX machine, you can access the same service using the nslookup command. Simply type a name like "www.howstuffworks.com" into the command line, and the command will query the name servers and deliver the corresponding IP address to you.

So here it is: The Internet is made up of millions of machines, each with a unique IP address. Many of these machines are server machines, meaning that they provide services to other machines on the Internet. You have heard of many of these servers: e-mail servers, Web servers, FTP servers, Gopher servers and Telnet servers, to name a few. All of these are provided by server machines.

URL

An acronym for "Uniform Resource Locator," this is the address of a resource on the Internet. World Wide Web URLs begin with <http://>

Example: the URL for the Department of Education home page is: <http://www.doenets.lk.htm>

This is the standard way to give the address of any resource on the Internet that is part of the World Wide Web (WWW).

This is the unique identifier, or address, of a web page on the Internet. It provides a way of uniquely specifying the address of any document on the Internet.

The typical URL specifies the method used to access the resource (the protocol), the name of the host computer on which it is located, and the path of the resource, eg : The protocol specified in this example is http (HyperText Transfer Protocol), the protocol of the World Wide Web. URL includes the type of naming scheme employed (http, ftp, telnet, news, file, etc.), a separating colon, the location of the host, and a path to the resource. URLs may be either absolute (containing the entire address of the resource) or relative (containing only a part of the address).

IP Address

Each machine connected to the Internet has an address known as an Internet Protocol address (IP address). The IP address takes the form of four numbers separated by dots.

Network addresses are usually of two types: (1) the physical or hardware address of a network interface card; for Ethernet this 48-bit address might be 0260.8C00.7666. The hardware address is used to forward packets within a physical network. (2) The logical or IP Address is used to facilitate moving data between physical networks and is made up of a network number, a sub network number, and a host number

Every machine that is on a network (a local network, or the network of the Internet) has a unique IP number [four sets of numbers divided by period with up to three numbers in each set. (ie 192.168.0.1)] - If a machine does not have an IP address it cannot be on a network. Most machines also have one or more Domain Names that are easier for people to remember.

The "address" or URL of a particular Web site. This is also how you describe the name that is at the right of the @ sign in an Internet address. For example, netlingo.com is the domain name of this Internet dictionary. There is an organization called InterNIC that registers domain names for a small fee and keeps people from registering the same name. Most recently, more domain names will be allowed due to new suffixes coming out.

Domain Name

A domain name is the text name corresponding to the numeric IP address of a computer on the Internet. A domain name must be unique. Internet users access your website using your domain name. download The process of copying files, information and images from the Internet to your computer. Every time a visitor accesses a page on the Internet, they are downloading the contents of that page.

Design Multimedia Contents for Websites

In this presentation you will be learning how to create an animation on a website
To create an Animation you can use applications like

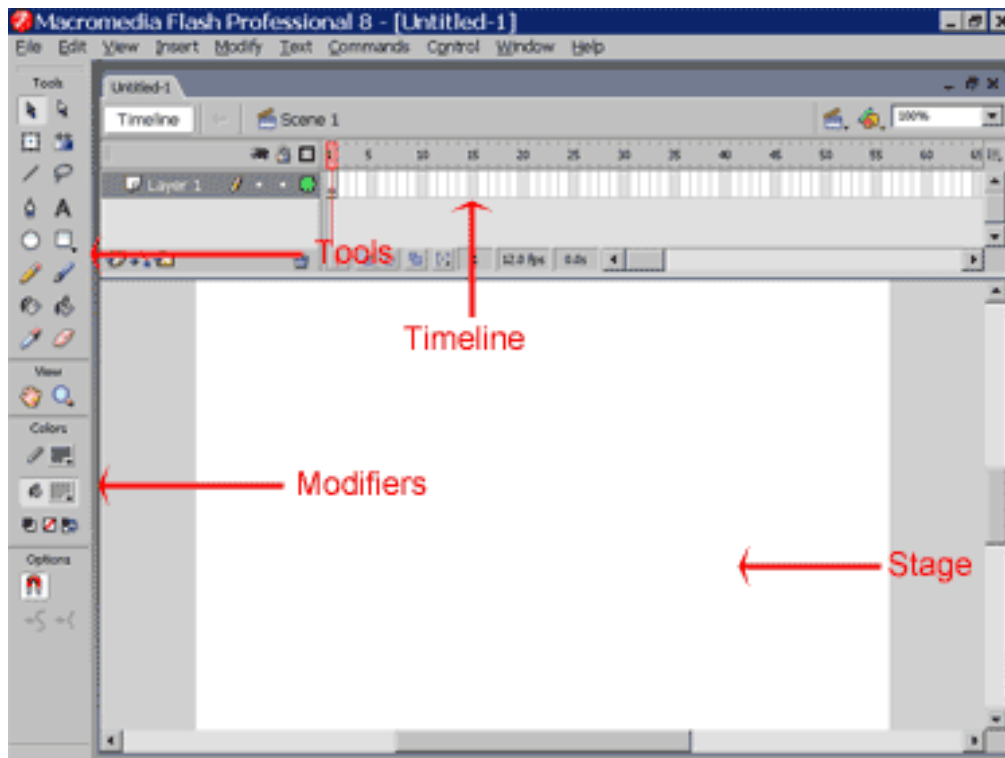
- PowerPoint
- Flash
- Moho

We will use Flash Professional 8

To begin, open Flash Professional 8. You will be presented with the screen shown here.



Click Flash Document. The screen shown appears below :



The upper left corner of the screen displays the Tools palette, which contains tools you can use to create or modify graphics and text.

The Stage is displayed in the center of the screen. You create your movie on the Stage.

The Timeline appears in the upper portion of the screen. You use the Timeline to lay out the sequence of your movie.

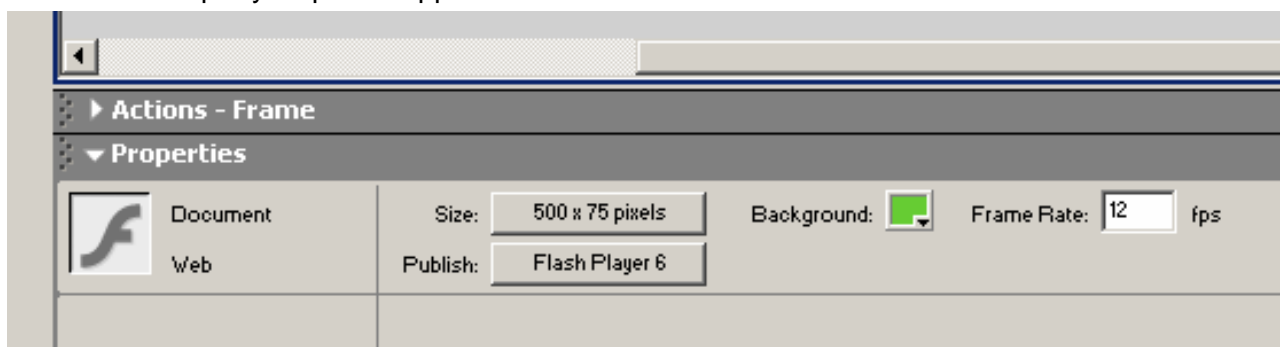
The Property Inspector

In the Property inspector, you can set the attributes of objects as you work. You will use the Property inspector frequently when working in Flash Professional 8.

To open the Property inspector:

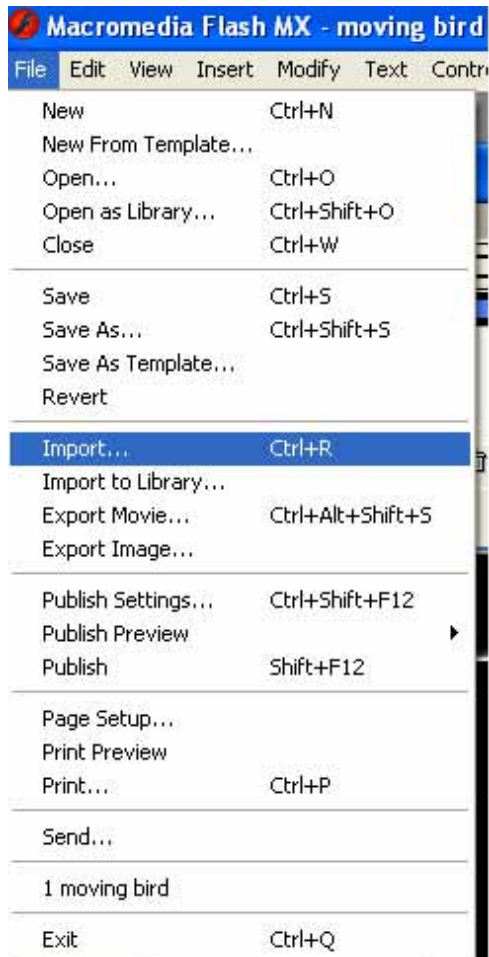
Choose *Window > Properties*

The Property inspector appears at the bottom of the screen

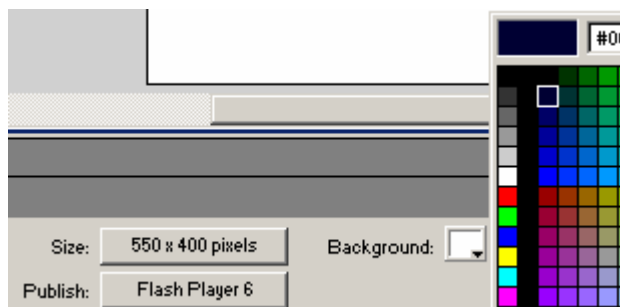


Create your animation using Flash

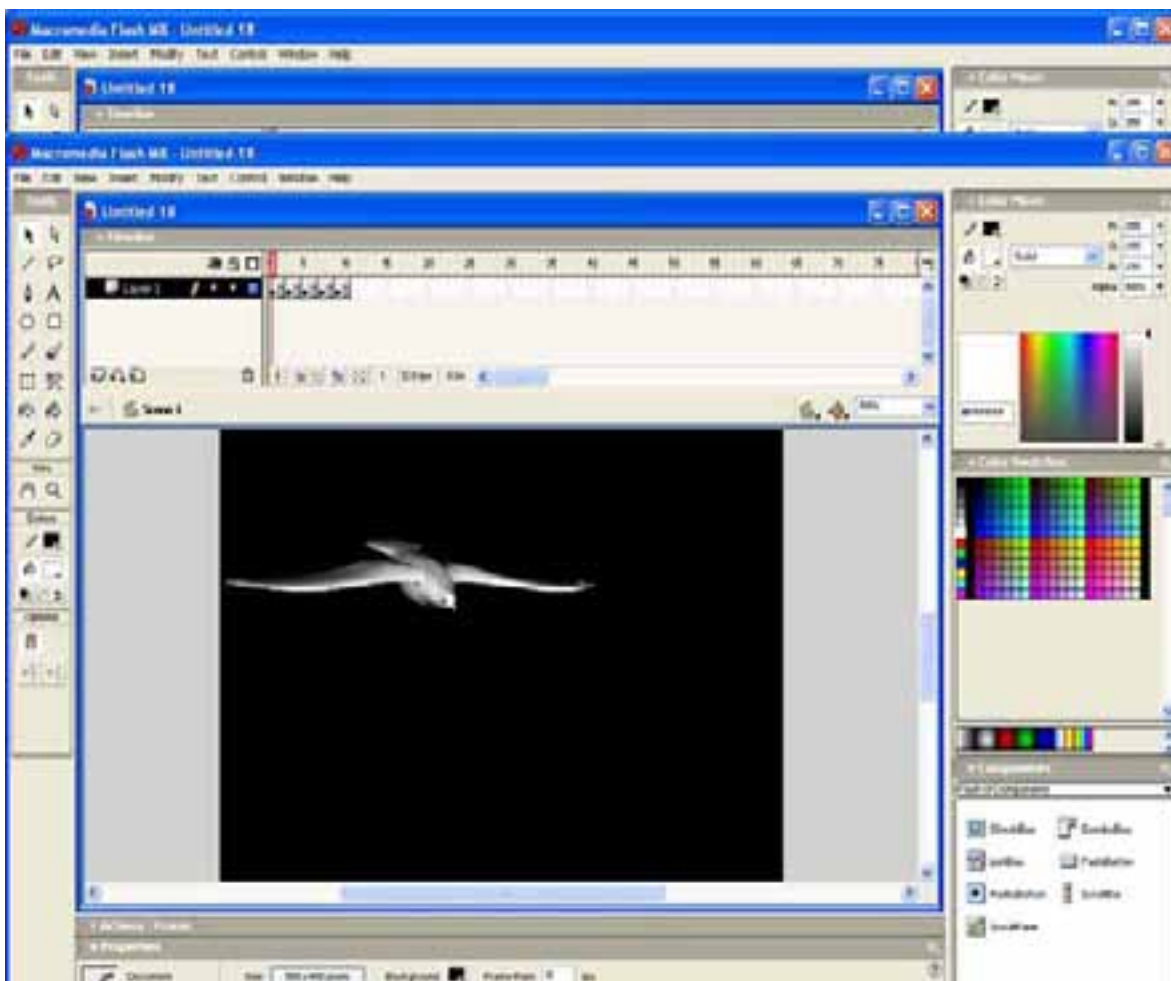
- Open Flash
- *File* → *New*
- *File* → *Import* and import moving *Bird.gif*



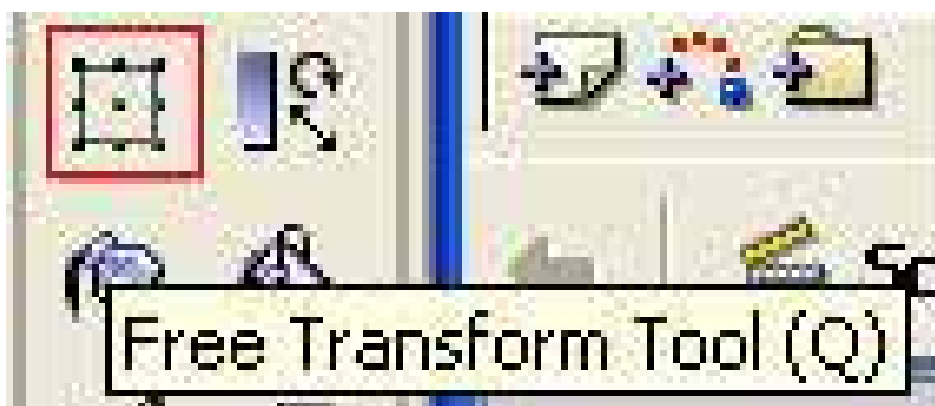
Change the background color to black



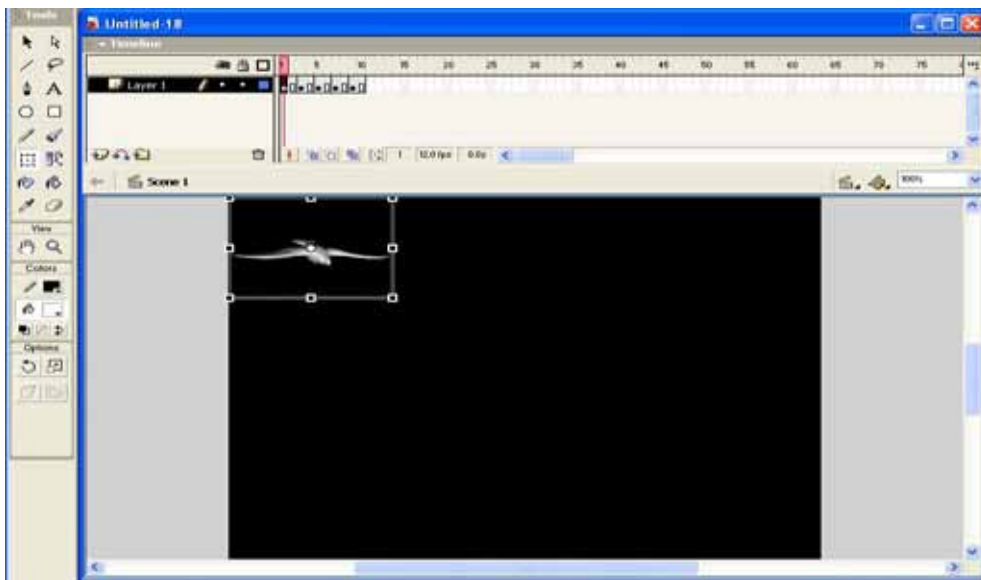
- Insert 5 key frames
It shown below



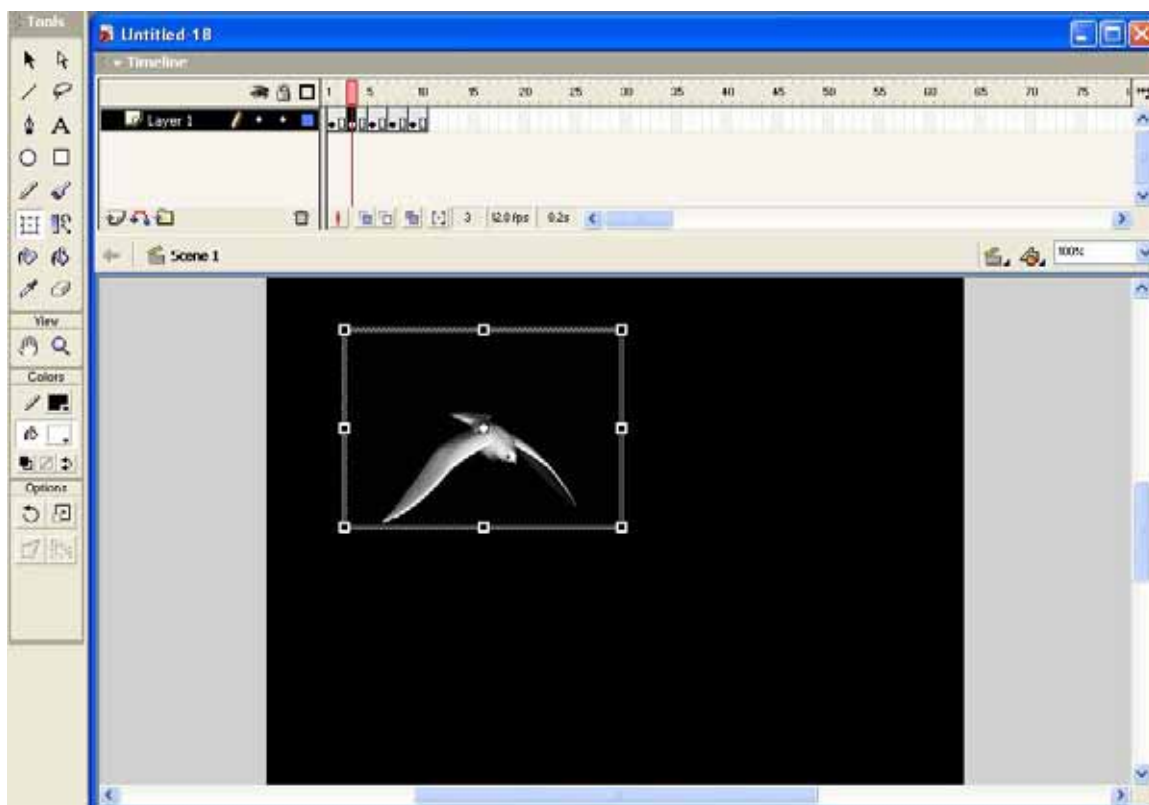
- Then click frame1 and click on the image
- Click Free Transform Tool.
- Now you can edit the size of the picture



After editing frame 1

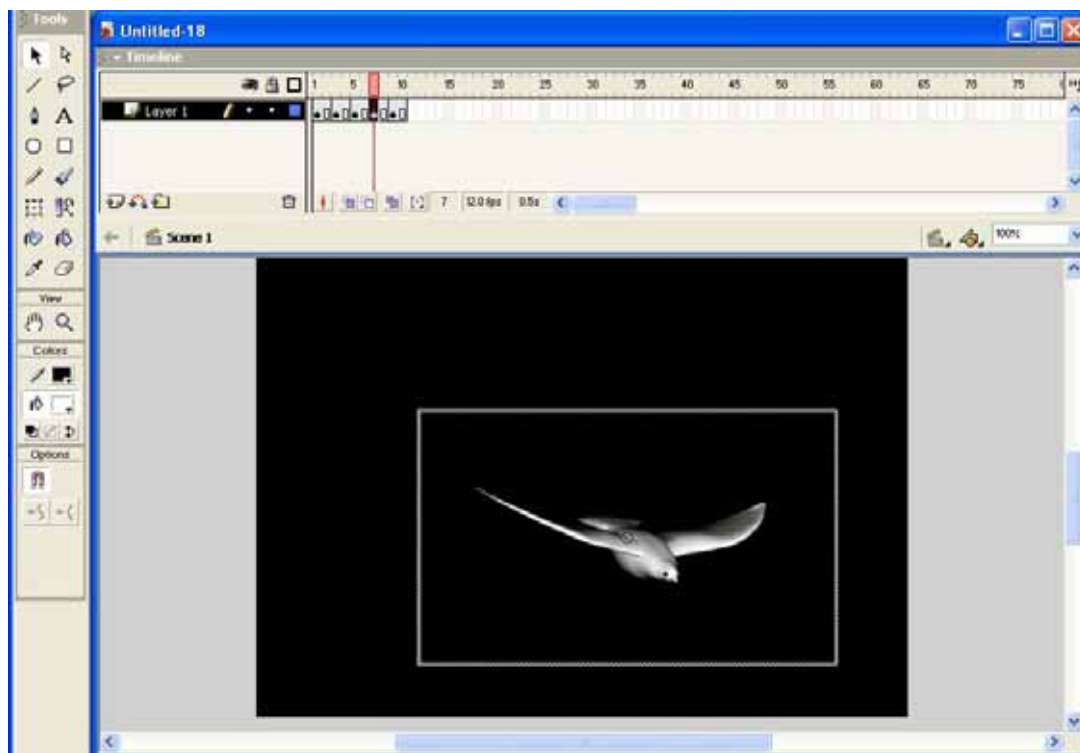
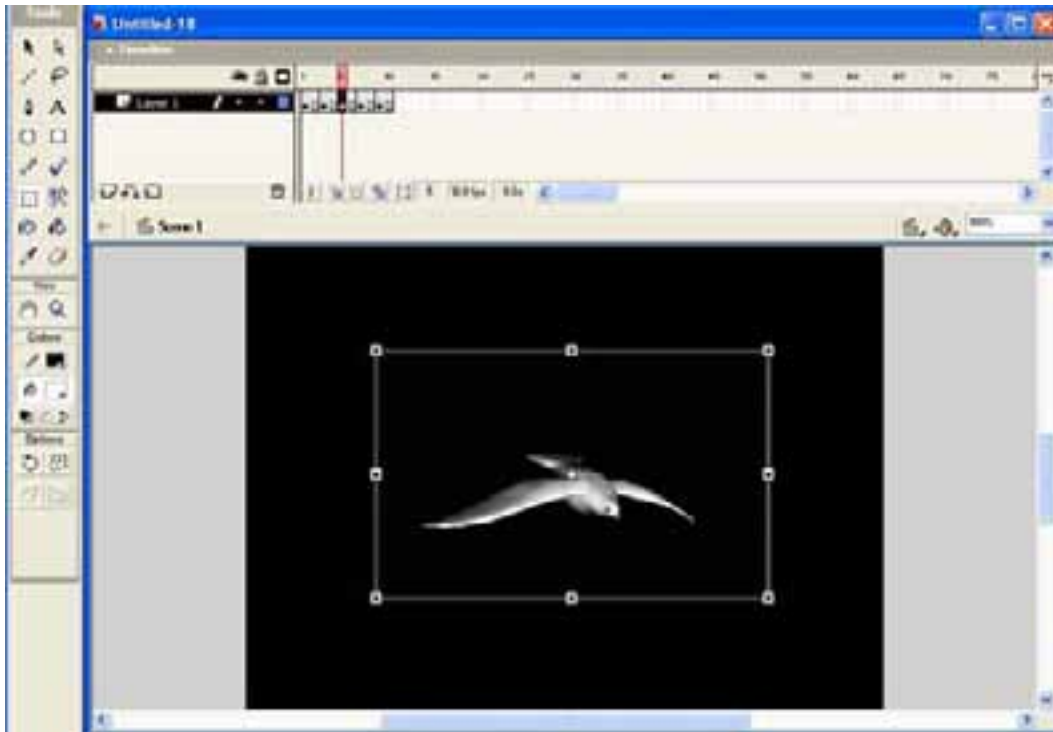


- Then go to frame 2 and edit picture as below by using *Free Transform Tool*.
- Click on the image and put mouse pointer inside the selected frame.
- Mouse pointer changes to a cross. Now you can move the picture.

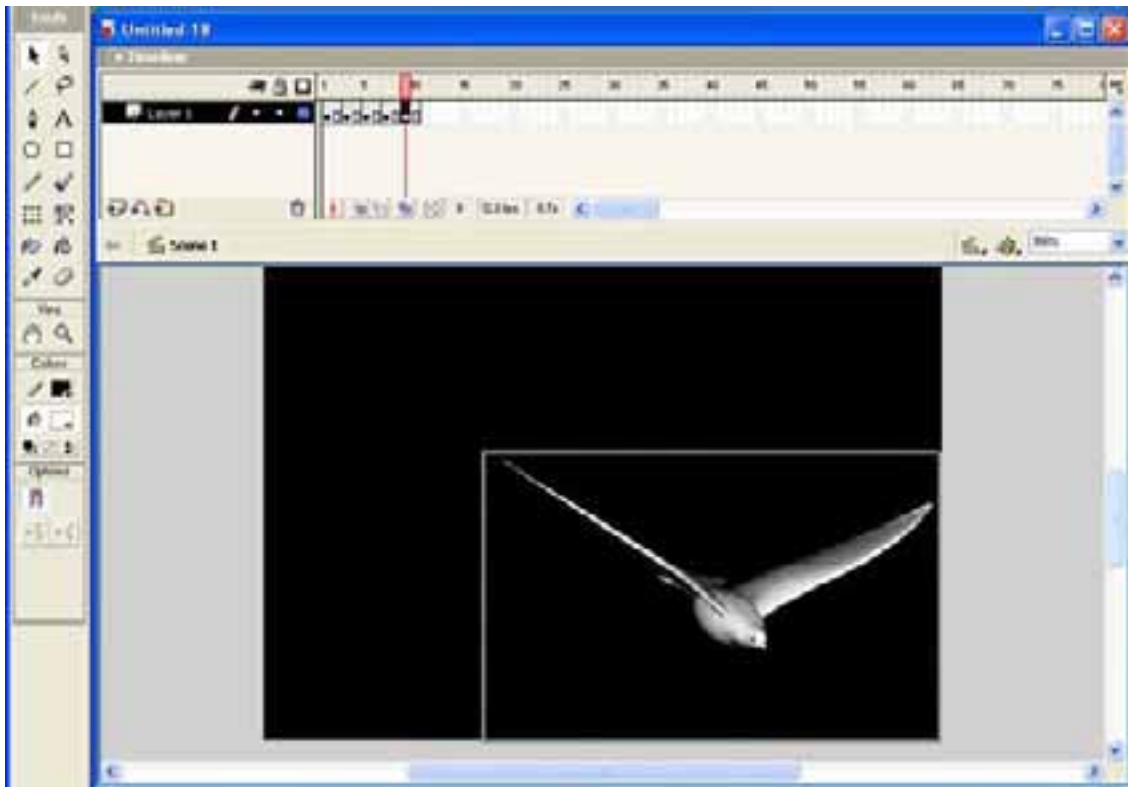


- Edit other key frames as above and create the animation.

Frame 3



Frame 5



Finally press *Ctrl+Enter* to view the animation

Create the flash movie

- *File* → *Export movie*
- Type the name *Bird* then click save
- Press *Ok*

The following code is used to create the web page

```
<head>
<title>Unit6. web Site Development</title>

</head>

<object>
<embed src="Bird.swf" width="550" height="400">
</embed>
</object>
```

Coding Audio and Video in HTML

The simplest way to call a sound file is to simply put a link to the file. Depending on your browser and configuration, it will either invoke your operating system's basic player software, player support built into the browser, or a plug-in that provides the player capability.

As an example of the simplest format, the line of HTML code below will call up the file *Track 2.mp3* from the current directory with any browser. Note that you can play any sound file with this command format, not just MIDI files:

```
<a href="Track 2.mp3"> Play My Song </a>
```

The simplest format of the line of HTML code will call up the file *GLOBE.AVI* from the current directory

```
<EMBED SRC="GLOBE.AVI" WIDTH=500 HEIGHT=500 >
```

Photoshop: Designing Graphics for the Web

- Photoshop Basics

- What is Photoshop?

- Uses

- Tour

- Web Graphics

- Graphic Limitations

- Display Considerations

- Cross-Platform/Browser-Safe Palette

- File Formats

- Transparent Text

- Saving for the Web

- Scanning

- Scanning Concepts

- Evaluating Originals

- Scanning Line Art vs. Photographic Images

- Scanning Strategies

- Photoshop Measurement/Correction Tools

- Additional Resources

What is Photoshop?

Photo retouching, image editing, and color painting program; graphic design tool

- Create high-quality digital images

- Tools & special effects capabilities

- Manipulate scanned images, slides, & original artwork

- Isolate parts of an image for experimentation & individual editing

- And lots more.....

Uses of Photoshop

- Art (line drawings, charcoal, color original)
- Photographic
- Restoration
- WWW (GIFS, JPEGS, etc.)
- Montage
- Halftones, Duotones, Tritones, Quadtones
- Color Separations
- Posterizations
- Special Effects

Touring Photoshop

- Using Help
- Navigating: Windows, Palettes, Features & Tools
- Preferences

Graphic Limitations

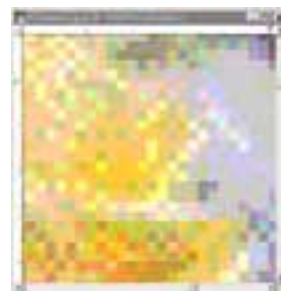
- Connection Speeds
- User Configurations

Display Considerations

- Screen Sizes
- Colors

Cross Platform/Browser-Safe Colors

- 256 vs. 216 Colors
 - Dithering
 - Using the Color Picker



File Formats

- JPEG – Joint Photographic Experts Group
- GIF – Graphics Interchange Format
- PNG – Portable Network Graphics

JPEG

- Best for photos or continuous tone, full-color images
- Uses 16 million colors
- Browsers use reasonable approximations
- Work in RGB mode
- Uses glossy compression
- Saving (Standard, Optimized, Progressive)

GIF

- Best for solid color images (buttons, logos)
- Uses 256 colors
- Browsers uses 216 colors
- Work in Indexed mode
- Good compression
- Interlaced

PNG (8-bit)

- Best for line art (logos)
- Compresses solid areas of color well and maintains sharp detail
- Generally, has better compression than GIF (10-30% smaller)
- If considering saving as GIF, also consider saving as PNG (8-bit)
- Not supported by older browsers

PNG (24-bit)

- Best for continuous-tone images
- Compresses well, but can be larger than JPEGs
- If considering saving as JPEG, could also consider PNG (24-bit)
- Not supported by older browsers

JPEG vs PNG



**10.8K PNG-8 with 64
GIF vs PNG**

9.5K GIF with 64



68K

31K JPG

Web Graphics

Saving for the Web

- 4-Up View
- File Sizes and Download Speeds
- Changing Options
- Halo Effect and Matte Options

Scanning Concepts

- Getting Images Into Photoshop
- Bitmap vs. Vector Graphics
- Pixel Dimensions
- Image Resolution
- Monitor Resolution
- Printer Resolution
- Screen Frequency
- File Size

Evaluating Originals

- Color Range
- Contrast
- Size
- Good Original
- Shadow & Highlight Detail
- Tonal Corrections

Scanning Line Art

Scan Resolution = Output Device Resolution X Sizing Factor

Scanning Photographic Images

Scan Resolution = Screen Ruling x Quality Factor x Sizing Factor

Sizing Factor = Desired Size/Original Size

Basic Image Correction

- Examine the Original
- Scan the Original
- Identify the Image Colors & Tones
- Measure & Adjust Highlights
- Measure & Adjust Shadows
- Measure & Adjust Midtones
- Apply Unsharp Mask
- Save File
- Review Image for Rescanning or Additional Corrections

Color & Tonal Adjustments

- Histograms
- Navigator Palette
- Gamma Settings
- Curve Controls
- Unsharp Mask Filter

Unit 07

ICT and Society

7.1 ICT for National Development

Introduction to Information and Communication Technology (ICT) and Society

In Sri Lanka, the government takes action to promote ICT development in order to widen the opportunities to cater to the under-privileged majority of the community.

This unit will provide an introduction to the relationship between ICT and Society. It assumes that we cannot understand ICT without reference to the concept of 'society', a term that requires careful theoretical reflection. The unit therefore explores economic, social and cultural theories about ICT, and encourages students to think about ICT from a variety of perspectives.

Globalization

Globalization is the process of minimizing or contracting the entire world into the size of an early village. This means that the world has merged through information. The entire world has become a global village with respect to communication. Now, with the introduction of modern ICT tools, any piece of information or news would reach any part of the world in a very short time.

In Sri Lanka, there are a number of programmes launched by the government to promote ICT development in order to widen the opportunities to cater to the underprivileged majority of the community.

e-Sri Lanka programme

The e-Sri Lanka initiative uses Information and Communication Technologies (ICT) to develop the economy of Sri Lanka, reduce poverty and improve the quality of life of the people.

The Sri Lankan government uses Information and Communication Technology (ICT) for

- developing the economy of Sri Lanka,
- reducing poverty and improving the quality of life of the people ,
- building information infrastructure and an enabling environment,
- developing ICT human resources,
- modernizing government and delivering services to citizens,
- developing society,
- ensuring equitable distribution of opportunities and knowledge.

ICT is constantly changing and advancing as scientists and engineers new technologies for us to use and

For example, in the last 50 years these



create
enjoy.

technologies have become common.

- Personal Computers
- Mobile Phones
- The Internet
- Medical scanners
- Satellites
- Lasers
- CD and DVD
- Television
- Automobile electronics
- ATM facilities
- Credit cards

e-Governance & e-Government

e-Governance and e-Government have come increasingly into focus as ways of strengthening governments and good governance. If implemented strategically e-governance can not only improve efficiency, accountability and transparency of government processes, but it can also become a tool to empower citizens by enabling them to participate in the decision-making processes of governments.

e-Government services for the poor assist governments in reaching the yet 'unreached' and thereby contribute to poverty reduction in rural and remote areas. At the same time, this process also enables involvement and empowerment of marginalized groups through their participation in the political process.

Despite the developmental potential, few governments in the Asia-Pacific region have planned for and implemented e-government strategically and directly targeting poor people. This is despite the fact that more than 60 percent of the population in the Asia-Pacific region live in rural areas where the majority of them are poor.

7.1.1 ICT in Health

With the development of ICT the health sector has developed in various aspects.

- It has made the life of the patient easier by facilitating channeling of the doctor from home
- For doctors, diagnosis and medical testing have become easier and reliable
- ICT collaborates with the administrations (hospitals, local and regional health organizations) in projects to computerize hospital and territorial services by sustaining strategic initiatives to reach the objectives of the National Health Plan.

ICT can be used to design and develop medical teleconsulting products targeted at providers of second opinion services. In the past twenty years, ICT has fundamentally changed both practice and medical education. Large hospitals are completely dependent on ICT systems to run - accounts, digitized radiography, laboratory ordering, processing and reporting systems, electronic patient record systems, etc., that have now become an integral part of the hospitals function. All laboratories are computerized, and it is reckoned that the failure of laboratory computer systems alone would cause large tertiary referral hospitals to grind to a halt in less than two days. Digital radiography started with the Computer-Aided Tomography (CAT) scan. Magnetic Resonance Imaging (MRI) and general digital imaging are now widespread, and film is becoming obsolete in radiology. Digital imaging has enormous advantages - images can be processed and improved and

analyzed: they can also be viewed quickly and at a number of locations within a hospital, or even at remote locations.

ICT has changed the way we access information in Medicine. Twenty years ago, to get information on the published material on a topic would necessitate a few days in the Medical Library . Now, in a few minutes, one can access an online Medline Database (available free!) and in minutes have the search completed back many years, have abstracts of the papers published on the computer desktop with a very simple method of ordering (or in some cases downloading directly) papers required.

The present, not to mention the future, of medicine, is digital.

Medical schools in this part of the world do not appear kept abreast of these technological changes.



to have

Computer simulation can reduce the need for animal testing. of a computer simulation to investigate saves time, money – lives of animals.

The use and the

Some systems enable clinicians to interpret three-dimensional x-ray pictures together – at their respective hospitals via broadband Internet.

Some systems are being used to monitor the progress of patients who have undergone surgery for, say, a thoracic aortic aneurysm. The elderly patients are saved the tedium of traveling for follow-up consultations. The system has been tested on patients, and it meant they do not need to make the long journey to the hospital for post-operative consultations. Now they can stay in one place and be checked from afar. Saving is also made on reduced hospital bed nights.

The electronic medication card sends automatic messages between the patient's physician and the local hospital, or the pharmacy, or the home care services using a complete electronic system for the exchange of prescription information.

Implants cure snoring and the sound of respiration. Three little pieces of plastic from a material called Dacron - operated into the palate cures 70 per cent of the patients subject to severe snoring. With this method there is not only any need for unpleasant surgery, but it also requires a shorter period of convalescence.



A new and revolutionary series of antibiotics - aimed at HIV, cancer and organ transplant medicine, are well under way.

7.1.2 ICT in Education

ICT in the education sector both ensures that schools have access to ICT but that education is enhanced with ICT. ICT can enhance education content, teacher training, and technology skills development and can reduce cost and location barriers in providing these services.



ICT has made major contributions in the areas of;

o Teaching

- Supporting teacher professional development through computer-mediated networks that facilitate tele-collaboration
- Using technology to break down entrenched rote-based teaching practices and to support educational reform towards more student-centered learning
- Using computers in classrooms for on-line access to the latest textbooks and teaching materials
- Integrating broadcast technologies, such as interactive radio instruction (IRI), TV and video programs, with digital technologies to improve delivery and content of teaching and learning activities

o Learning

- Developing networks that promote student collaboration both within the country and with students abroad

Learning Management Systems

- A Learning Management System (or LMS) is a software package, usually of a large scale (that scale is decreasing rapidly), that enables the management and delivery of learning content and resources to students. Most LMSs are web-based to facilitate "anytime, anywhere" access to learning content and administration.
- At a minimum, the LMS usually allows for student registration, the delivery and tracking of e-learning courses and content, and testing, and may also allow for the management of instructor-led training classes. In the most comprehensive of LMSs, one may find tools such as competency management, skills-gap analysis, succession planning, certifications, virtual live classes, and resource allocation (venues, rooms, textbooks, instructors, etc.). Most systems allow for learner self-service, facilitating self-enrollment, and access to courses.

o Educational Management

- Developing education management information systems (EMIS) to improve the administration of the education setting
- Learning Management Systems(LMS)



An International corporation located in the UK may want to train their staff located in Sri Lanka on a new computer application. Normally, the staff would have to come to the UK for training. But now, the UK office can set up a video link with the office, they also set up remote control of the PCs in Sri Lanka and they run the training course directly from the UK. Everybody wins.

7.1.3 ICT In Agriculture

ICT in the agriculture sector facilitates knowledge sharing within and among a variety of agriculture networks including researchers, importers/exporters, extension services, and farmers. ICT enables vital information flows by linking rural agricultural communities to the Internet, both in terms of accessing information and providing local content.



7.1.3 ICT activities in agriculture :

- Use of Internet and e-mail for extension purposes
- Communicating agro-meteorological information
- Communicating market price information
- Facilitating networks of agriculture researchers
- Developing land registries

In addition to Internet as the backbone, the province run television station, call center, telephone, mobile phone, and village-run broadcast will be used to meet farmers' needs using so-called "all-round ICT service". It will be a participatory approach to the development strategy of information service. All stakeholders should be mobilized to contribute their money, labor, or knowledge including government agencies, private sectors, companies, farmers, marketers, technicians and professionals with agricultural information and knowledge.

7.1.4 ICT in Industries

ICT contributes to the rapid development of various other industries.

This is through three main areas viz:

- Computer Aided Design (CAD)
- Computer Aided Manufacture (CAM)
- Robotics



In many industries the traditional drawing board is relic. There are many **computer aided design** packages that enable the designer to work out the Drawing with a mouse is difficult, so the computer may have:

almost a
(CAD)
designs.
system

- a light pen;
- a graphics tablet;

In this picture you can see that under the engineer's left hand there is a **graphics tablet**.



The advantages of CAD are many:

- no special drawing skills are needed;
- designs can be easily changed.
- the design can be tested using **computer modeling** . Complex mathematical calculations are used to predict the item's behaviour under normal and extreme conditions
- plans can be stored.

The computer design, when complete,

can be loaded into a **computer aided manufacturing (CAM)** machine.

An example of this is seen in the picture below.

Such machines are massively expensive, but pay for

- greater productivity;
- better precision, leading to greater reliability finished product.
- permitting production of complex shapes.



themselves in:

of the

Many assembly lines are now operated by **robots** controlled by computers.

Robots are complex machines that rely on feedback



7.1.5 ICT in other Services

7.1.6

Other than the sectors mentioned above, ICT has catered to the development of various other services such as Banking, Tele Commuting, Insurance, Stock Market etc. .

Tele Commuting

Being able to access the company network from anywhere means that people are no longer tied to the office, they could just as easily work from home. Because of this, home working ('Teleworking') is becoming more common.

Also, people working for international corporations can travel from country to country on business and yet settle down to a fully networked local office desk and work as if they were in their home office



Friends and family , no matter where they are on the planet, people can talk to one another if they have access to the right technology. For example, there is a very busy Internet cafe set up at the base camp of Mount Everest

Travel and the Environment: Video conferencing and email have reduced the need for business travel. This has allowed people to have more time at home with their families rather than being stuck in an airport somewhere. Less travel also means less pollution, as fewer cars and aircraft need to be used.



Entertainment

ICT is everywhere in the modern homes. It has affected the way we go about our daily lives.

Multi-channel digital television

There are so many transmission channels now available, that broadcasters create special interest channels to attract an audience e.g. Sports, Science, Cooking, Travel and so on

Computer games

In recent times, computers have become so powerful and low cost that many of us play computer games for entertainment. You can buy specialized games machines that hook up to a television.

or you can use the Internet to play online games with thousands of other people who are also online with you.

Music.

Earlier there were vinyl records and the radio. There were no affordable recording methods. Today we listen and gather our music in a number of new ways.

The invention of the Compact Disc made a huge leap in sound quality .We are able to record our music straight on to a computer. We can pay for a music track in an online music store and download it to our personal MP3 player.

Family Life - Generation Gap

ICT technology moves at such a pace that older generations may be 'left behind'. Today, ICT is as a core subject in schools, so that technology is familiar to school children. Not so for older people, have less opportunity to use the technology available.



taught
very
who



'Silver surfers'

Those older people who have embraced the internet and made it part of their life are called 'silver surfers' by the media.

For example, some people have a serious illness to cope with, such as cancer. And they have turned to online diaries called 'web logs' to share their experiences with others.

Online shopping

Online shopping is said to be one of the most popular uses. We 'll be able to do many chores online if we wish. For example food shopping is simple with most of the big supermarkets having an online store. Once an order is placed, it will be delivered by van

within an agreed time slot.. Other specialist shops can supply organic food and direct-from-farm produce.

World Awareness

The 24 hour news networks brings us events from around the world as they happen. This means that as a Society we can react almost immediately. In natural disasters such as the Boxing day Tsunami, massive aid from nations from around the world was brought to bear within hours. Wars, crimes, tragedies, celebrations are much closer to us than they were 50 years ago.

Social connections:

People can turn to the Internet when they need career advice, helping people through an illness or finding a house. It shows that the Internet has become a cornerstone when searching for vital information.



new

ICT in banking

- Visa card
- Master card
- Credit card
- ATM cards

An **automated teller machine** or **automatic teller machine (ATM)** is an electronic computerized telecommunications device that allows a financial institution's customers to directly use a secure method of communication to access their bank accounts, order or make cash withdrawals (or cash advances using a credit card) and check their account balances without the need for a human bank teller (or *cashier* in the UK). Many ATMs also allow people to deposit cash or cheques, transfer money between their bank accounts, top up their mobile phones' pre-paid accounts or even buy postage stamps.



On most modern ATMs, the customer identifies him or herself by inserting a plastic card with a magnetic stripe or a plastic smartcard with a chip, that contains his or her account number. The customer then verifies their identity by entering a passcode, often referred to as a **PIN (Personal Identification Number)** of four or more digits. Upon successful entry of the PIN, the customer may perform a transaction.

If the number is entered incorrectly several times in a row (usually three attempts per card insertion), some ATMs will attempt to retain the card as a security precaution to prevent an unauthorised user from discovering the PIN by guesswork. Captured cards are often destroyed if the ATM owner is not the card issuing bank, as non-customer's identities cannot be reliably confirmed.

7.2 Issues in the use of ICT

7.2.1 Ethical , Legal and social issues

Computers involve a special technology and they raise some special ethical issues. Computer ethics is the analysis of the nature and social impact of computer technology and the corresponding formulation and justification of policies for the ethical use of such technology. It concerns software as well as hardware and concerns networks connecting computers as well as computers themselves.

The Internet provides access to a variety of information on every topic and this information comes from many different countries throughout the world. One problem with the Internet is that all the information is freely available once a user is connected. There are areas of the Internet, which contain large amounts of illegal material. Material that is illegal in some countries may be perfectly legal in others.

Governments have the problem of finding a way of allowing users to gain access to the Internet but not to any illegal areas. If access to such material is restricted on one part of the Internet a user can simply move to another area to find a way to access the material.

There is a problem in restricting access. The Internet is a global system and it is difficult for single countries to make laws to control it. Another problem with restriction is that it could lead governments to begin attempts to censor, legislate and regulate the Internet for political, cultural and religious reasons. Civil liberty groups are naturally concerned about this aspect of control.

Computers and Privacy

The rapid explosion in the use of computers in the last 15 years has benefited us in many ways. Many things that we now take for granted, such as the use of credit cards and cash dispensers would have been impossible without them. However, there are problems. As more computers are used, more and more information about each of us is stored on computers. By linking the information gained from several computers together it is possible to build up a complete picture of a person's life.

Software theft

It could be said that the use of personal computers has turned many users into thieves. How many people could honestly say that all the software on their hard disks has been purchased by them? As you can see from the Copyright, Designs and Patents Act 1989, it is a criminal offence to copy or steal software.

Hacking

Hacking means gaining illegal access to someone else's computer system. Many people see this type of thing as a challenge and not as an illegal activity.

Privacy

As more and more information is held there is the chance of some of it being incorrect. Your private life is becoming less and less private.

Job losses

Is it right to develop new systems in the knowledge that staff will inevitably be made redundant? Should we put shareholders' dividends and profits before people? These are difficult questions and ones which need to be addressed. Everyone has his own opinion on this. What is yours?

Digital Divide (Social Exclusion)

Digital divide is the division of people according to the accessibility to ICT Resources. There is a gap between those who can access ICT resources easily and those who cannot.

The digital divide results from the socio-economic differences between communities that in turn affects their access to digital information mainly but not exclusively through the Internet. Broadly speaking, the difference is not necessarily determined by the size or depth of the user group. Any digital medium that different segments of society can use, can become the subject of a digital divide. The digital divide is not a clear single gap that divides a society into two groups. Researchers report that disadvantages can take such forms as lower-performance computers, lower-quality or high-priced connections (i.e. narrowband or dialup connections), difficulty in obtaining technical assistance, and less access to subscription-based content.

7.2.3 Security issues

Physical security

Environmental factors

Your computer should be kept in a

- Dust free
- Dry (moisture free)
- Smoke free environment.

Hardware protection

Your computer should be equipped with

- A UPS(Uninterrupted Power Supplier) to avoid risk of sudden power failure and fluctuations.
- Surge protection to protect against lightening and thunder.
- Stabilizer to control voltage.

Logical Security

The software and the data in your computer can be protected through the use of

Passwords
Backups

Malicious codes

A common misconception is that other kinds of electronic nasties such as worms and Trojan horse applications are viruses. They aren't.

Worms, Trojan horses, and viruses belong to a broader category analysts call "malicious codes."

- Virus
- Worm
- **Trojan Horse**

Virus

A program or piece of code that is loaded onto your computer without your knowledge and runs against your wishes. Like any other program, it contains instructions that tell your computer what to do.

- Viruses can also replicate themselves.
- All computer viruses are manmade.
- A simple virus that can make a copy of itself over and over again is relatively easy to produce.
- A virus can be very destructive; it could format your hard drive, overwrite your hard drive boot sector, or delete files and render your machine inoperable.
- Even such a simple virus is dangerous because it will quickly use all available memory and bring the system to a halt.
- An even more dangerous type of virus is one capable of transmitting itself across networks and bypassing security systems.

How do we get Viruses?

- Viruses enter your system via;
 - e-mail or an e-mail attachment
 - downloads
 - shared infected floppy disks
 - (occasionally) hacking

How Viruses work?

Once you open an infected file or application, the malicious code copies itself into a file on your system, where it waits to deliver its payload -- whatever the programmer designed it to do to your system. Simply deleting the e-mail after you open the attachment won't get rid of the virus, since it has already entered the machine.

A virus writer can set the payload to trigger immediately, at a preset future time or date, or upon the execution of a specific command, such as when you save or open a file. The Michelangelo virus, for example, was programmed to release its payload on March 6 of any year -- the artist's birthday.

The Difference Between a Virus, Worm and Trojan Horse

Viruses, worms and Trojan Horses are all malicious programs that can cause damage to your computer, but there are differences among the three, and knowing those differences can help you to better protect your computer from their often-damaging effects.

Computer Virus

A **computer virus** attaches itself to a program or file so that it can spread from one computer to another, leaving infections as it travels. Much like human viruses, computer viruses can range in severity: Some viruses cause only mildly annoying effects while others can damage your hardware, software or files. Almost all viruses are attached to an executable file, which means the virus may exist on your computer but it cannot infect your computer unless you run or open the malicious program. It is important to note that a virus cannot be spread without human action, (such as running an infected program) to keep it going. People continue the spread of a computer virus, mostly unknowingly, by sharing infecting files or sending e-mails with viruses as attachments in the e-mail.

General Virus Types

While there are thousands of variations of viruses, most fall into one of the following six general categories, each of which works its magic slightly differently:

Boot Sector Virus

Boot Sector Virus replaces or implants itself in the boot sector---an area of the hard drive (or any other disk) accessed when you first turn on your computer. This kind of virus can prevent you from being able to boot your hard disk.

File Virus

File Virus infects applications. These executables then spread the virus by infecting associated documents and other applications whenever they're opened or run.

Macro Virus

Macro Viruses are written using a simplified macro programming language. These viruses affect Microsoft Office applications such as Word and Excel, and account for about 75 percent of viruses found in the wild. A document infected with a macro virus generally modifies a pre-existing, commonly used command (such as Save) to trigger its payload upon execution of that command.

Multipartite Virus

Multipartite virus infects both files and the boot sector--a double whammy that can re infect your system dozens of times before it's caught.

Polymorphic Virus

Polymorphic Virus changes code whenever it passes to another machine; in theory these viruses should be more difficult for anti-virus scanners to detect, but in practice they're usually not that well written.

Stealth Virus

Stealth Virus hides its presence by making an infected file not appear infected, but doesn't usually stand up to antivirus software.

Worm

A **worm** is a special type of virus that can replicate itself and use memory, but cannot attach itself to other programs.

A worm program replicates itself and slithers through network connections to infect any machine on the network and replicate within it, eating up storage space and slowing down the computer. But worms don't alter or delete files.

A **worm** is similar to a virus by its design, and is considered to be a sub-class of a virus. Worms spread from computer to computer, but unlike a virus, it has the capability to travel without any help from a person. A worm takes advantage of file or information transport features on your system, which allows it to travel unaided. The biggest danger with a worm is its capability to replicate itself on your system, so rather than your computer sending out a single worm, it could send out hundreds or thousands of copies of itself, creating a huge devastating effect.

One example would be for a worm to send a copy of itself to everyone listed in your e-mail address book. Then, the worm replicates and sends itself out to everyone listed in each of the receiver's address book, and the manifest continues on down the line.

Due to the copying nature of a worm and its capability to travel across networks the end result in most cases is that the worm consumes too much system memory (or network bandwidth), causing Web servers, network servers and individual computers to stop responding. In more recent worm attacks such as the much-talked-about, Blaster Worm., the worm has been designed to tunnel into your system and allow malicious users to control your computer remotely.

Trojan Horse

A **Trojan Horse** is full of as much trickery as the mythological Trojan Horse it was named after. The Trojan Horse, at first glance will appear to be useful software but will actually do damage once installed or run on your computer. Those on the receiving end of a Trojan Horse are usually tricked into opening them because they appear to be receiving legitimate software or files from a legitimate source.

A Trojan horse doesn't replicate itself, but it is a malicious program disguised as something benign such as a screen saver. When loaded onto your machine, a Trojan horse can capture information from your system -- such as user names and passwords--or could allow a malicious hacker to remotely control your computer.

When a Trojan is activated on your computer, the results can vary. Some Trojans are designed to be more annoying than malicious (like changing your desktop, adding silly active desktop icons) or they can cause serious damage by deleting files and destroying information on your system.

Trojans are also known to create a backdoor on your computer that gives malicious users access to your system, possibly allowing confidential or personal information to be compromised. Unlike viruses and worms, Trojans do not reproduce by infecting other files nor do they self-replicate.

Blended Threat

Added into the mix, we also have what is called a **Blended Threat**. A blended threat is a sophisticated attack that bundles some of the worst aspects of viruses, worms, Trojan horses and malicious code into one threat. Blended threats use server and Internet vulnerabilities to initiate, transmit and spread an attack. This combination of method and techniques means blended threats can spread quickly and cause widespread damage.

Characteristics of blended threats include: causes harm, propagates by multiple methods, attacks from multiple points and exploits vulnerabilities.

To be considered a blended threat, the attack would normally serve to transport multiple attacks in one payload. For example it wouldn't just launch a DOS attack — it would also install a backdoor and damage a local system in one shot. Additionally, blended threats are designed to use multiple modes of transport.

For example, a worm may travel through e-mail, but a single blended threat could use multiple routes such as e-mail, IRC and file-sharing sharing networks. The actual attack itself is also not limited to a specific act. For example, rather than a specific attack on predetermined .exe files, a blended threat could modify exe files, HTML files and registry keys at the same time — basically it can cause damage within several areas of your network at one time.

Blended threats are considered to be the worst risk to security since the inception of viruses, as most blended threats require no human intervention to propagate.

[Source: Symantec Security Response Web site]

Combating Viruses, Worms and Trojan Horses

The first steps to protecting your computer are to ensure your operating system (OS) is up-to-date. This is essential if you are running a Microsoft Windows OS. Secondly, you should have anti-virus software installed on your system and ensure you download updates frequently to ensure your software has the latest fixes for new viruses, worms, and Trojan horses. Additionally, you want to make sure your anti-virus program has the capability to scan e-mail and files as they are downloaded from the Internet. This will help prevent malicious programs from even reaching your computer. You should also install a firewall as well.

Antivirus software

Since 1987, when a virus infected ARPANET, a large network used by the Defense Department and many universities, many anti-virus programs have become available. These programs periodically check your computer system for the best-known types of viruses.

Antivirus software can detect nearly all types of known viruses, but it must be updated regularly to maintain effectiveness. Virus experts have recorded more than 40,000 viruses and their variant strains over the years, though only about 200 of those viruses are actively spreading in the wild. While most viruses are just annoying time-wasters, the ones that do deliver a destructive payload are a real threat.

Viruses have been around since the early 1960s, almost since the earliest computers existed, though until the 1980s they were largely laboratory specimens, created by researchers and released in a controlled environment to **examine their effect**. e-mail viruses can infect thousands of machines in a matter of minutes.

Firewall

A firewall is a system that prevents unauthorized use and access to your computer.

A firewall can be either hardware or software.

Hardware firewalls provide a strong degree of protection from most forms of attack coming from the outside world and can be purchased as a stand-alone product or in broadband routers.

Unfortunately, when battling viruses, worms and Trojans, a hardware firewall may be less effective than a software firewall, as it could possibly ignore embedded worms in outgoing e-mails and see this as regular network traffic.

For individual home users, the most popular firewall choice is a software firewall.

A good software firewall will protect your computer from outside attempts to control or gain access to your computer, and usually provides additional protection against the most common Trojan programs or e-mail worms. The downside to software firewalls is that they will only protect the computer they are installed on, not a network.

It is important to remember that on its own a firewall is not going to rid you of your computer virus problems, but when used in conjunction with regular operating system updates and a good anti-virus scanning software, it will add some extra security and protection for your computer or network.

Key Terms To Understanding Computer Viruses:

Virus

A program or piece of code that is loaded onto your computer without your knowledge and runs against your wishes.

Trojan Horse

A destructive program that masquerades as a benign application. Unlike viruses, Trojan horses do not replicate themselves.

Worm

A program or algorithm that replicates itself over a computer network and usually performs malicious actions.

Blended threat

Blended threats combine the characteristics of viruses, worms, Trojan Horses, and malicious code with server and Internet vulnerabilities.

Antivirus program

A utility that searches a hard disk for viruses and removes any that are found.

Practice safe computing

The best way to protect you from viruses is to avoid opening unexpected e-mail attachments and downloads from unreliable sources.

Resist the urge to double-click everything in your mailbox.

If you get a file attachment and you aren't expecting one, e-mail the person who sent it to you before you open the attachment. Ask them if they meant to send you the file, what it is, and what it should do.

For added safety, you need to install reliable anti-virus scanning software and download updates regularly.

Major anti-virus software vendors, including Symantec, Network Associates, Computer Associates, and Trend Micro, provide regular updates.

Some of the vendors also offer a service that will automatically retrieve updates for you from the company's Web site.

How anti-virus software works

Scanning software looks for a virus in one of two ways. If it's a known virus (one that has already been detected in the wild and has an antidote written for it) the software will look for the virus's signature -- a unique string of bytes that identifies the virus like a fingerprint -- and will zap it from your system.

Most scanning software will catch not only an initial virus but many of its variants as well, since the signature code usually remains intact.

In the case of new viruses for which no antidote has been created, scanning software employs heuristics that look for unusual virus like activity on your system. If the program sees any funny business, it quarantines the questionable program.

7.2.4 Health and Safety Issues

Computer habits can impact health

Using of computers continuously over a longer periods may cause of certain illnesses in your body. Some of them are;

- Eyestrain
- Back aches
- Uneasiness
- Tumors in the brain etc.

Are you bent out of shape with your computer? No, not because it won't print or because you have to keep pressing "control-alt-delete."

If you surf or type for hours without taking a break, you may literally be bent out of shape. In addition to frustration, the ills that come with prolonged computer use range from eyestrain to headaches to repetitive motion injuries to back pain.

There are three basic components of healthy computing: The physical environment, how you use the computer, and how you don't use the computer.



Workstation adaptation

In a perfect world, you'd sit in your very own comfortable and supportive chair to work on your large-screen computer in a well-lit room. In reality, you may be sitting in an office chair that's been handed down from someone six inches taller or shorter than you, squinting at a small, blurry computer screen. Given that, know that most of the critical elements of a healthy workstation - chair, desk, mouse, monitor, and lighting - can be tweaked to make them more user-friendly. The highlights:

- Your head should be about 18 - 30 inches from the monitor.
- The keyboard should be at elbow height, and at a distance and angle that allow you to keep your wrists straight.
- If your chair doesn't have arms, use a pillow to support your arms.
- Adjust your chair so that your feet are flat on the ground and your hips are slightly higher than your knees.

The Occupational Safety and Health Administration has a terrific site on how to set up the most important components of your workstation: <http://www.osha.gov/SLTC/etools/computerworkstations/components.html>.

Many larger workplaces have an ergonomic specialist as part of their occupational health or environmental safety offices. If a consultation is available to you, take advantage of it to improve your office workspace, and then take the tips home to use in your home office.

A special note for those of us who are approaching "a certain age." Computer users who wear bifocals or trifocals often tilt their head back to read through the bottom part of their glasses. This can cause neck, shoulder, and back problems. The Centers for Disease Control recommends lowering your monitor or using glasses designed specifically for computer work.

User modification

It's not all about the computer! If you sit like I do, pretzel-style, uncross your legs and get your left elbow off the desk so your shoulders are level. Speaking from experience, it's not easy to change the way you sit and work, but your musculoskeletal system will thank you. Proper sitting techniques for computer work:

- Keep your feet flat on the floor.
- Don't slouch.
- Keep your head in a "neutral" position

Shift your position slightly every so often to maintain good circulation and avoid stressing one area. If you have to look at books or printed documents while working, keep them close in front of you instead of twisting around to look at them.

And do you cradle the telephone between your shoulder and jaw while you're working? No good can come of this! If you must chat and work, either use a speakerphone or invest in a lightweight headset.

Operator exercises

Even if you have the perfect ergonomic set up and posture, you still need to take frequent breaks. "Repetitious static work," i.e., using the computer, is very fatiguing mentally as well as physically (although it's not the "good tired" that you get from physical activity). You can take a break from the computer without goofing off. Instead of leaving all the filing until the last hour of the day, intersperse it in 15 minute chunks. The same goes for making phone calls, etc.

Some experts recommend a three-minute break every half hour or so. In addition, those of us in the wellness world also recommend several breaks a day to stretch and use your larger muscles - like your heart! A few brisk, five-minute walks can leave you healthier as well as more refreshed and ready to tackle your work.

The body parts that often take the brunt of intense computer work are the eyes, neck, back, and hands. Some quick tips:



- Blink frequently so your eyes don't get dry.
- Clean the screen every once in a while so you're not peering through a layer of dust that has become one with your screen through the magic powers of static electricity.
- Refocus your eyes every ten minutes or so by looking away from the computer and briefly focusing on an object at least 20 feet away.
- Release neck strain by looking back over your shoulder while sitting up straight. Hold for 10 seconds, then slowly turn and look the other way.
- Relax shoulders by rolling them backwards and forwards.
- Reduce muscle tension in your hands by spreading your fingers wide apart. Hold, and then make a tight fist.

Unit 8.

Group Project

Project ideas for software systems

Introduction:

What is a project? :

Individual /group completed & well-documented project, is a clear demonstration of the students capability to handle /solve a common problem /task by using his knowledge & practical effort by using a computer, hardware and software environments available.

The aims of a project:

The aims of a completed project are to, Asses the demonstrated capability of the student in solving a common problem, to demonstrate the application of a computer to solve common problems by using the available software & hardware environments, & to demonstrate & explain ideas of the project in a standard documentation. Few such simple features can identify a project of this type; it should target to solve a common problem in the society, which may include a database, algorithms, hardware & software or menu systems, inputs & outputs with a set of given routines.

Unsuitable project topics:

For both computing and information technology projects, the system must involve a user inputting data of some kind, and obtaining output, the production of a documentation about some aspect of computing is **therefore not suitable.**

Examples:

- A project, which consists of a comparative study of two word processors (software comparisons).
- A project, of production of a magazine using a desktop publishing system (documentations only).
- A project, of general description of how to use a particular spreadsheet or database or database package (explaining common packages).

Suitable project topics:

Project work must run on a PC or related computer system with the available software & hardware with a clear & well-documented standard documentation.

It should target to solve a common problem in the society, which may include, a database, algorithms, hardware & software or menu systems, inputs & outputs with a set of given routines, with a clear & completed documentation.

1. Student Registration
 - a. Handling Facilities fees
 - b. Class Registration
 - c. Preparation of Marks sheet
 - d. Duty roster in the class room cleaning
2. Collection of data including Addresses of Friends
3. Collection of Rainfall data in student residential area. This data helps farmers to predict weather
4. Daily duty reports of teachers
5. Handling sports meet data, selecting students according to the houses, handling events, preparing reports

1. Preparation of activity plan materials
 - a. Identify shapes whether it is triangular or not,
 - b. Identify health conditions
2. Computer Lab Computer Booking System
3. Updating GCE O/L and A/L Results
4. Analysis of Sports meet Results
5. Maintaining records on home income expenditure
6. Word Pronunciation
7. Hardware Control - `Port programming
8. School Entrance Security System
9. Class Duty Chart
10. Collection of Herbal/medicinal Plants and other Rarely Available plants

1. Students Progress

- *Goals:*

- ⑩ Selection of students according to the leadership
 - ⑩ Identifying weaknesses of students
 - ⑩ Issuing character certificates
2. Preparation of health report according to daily health condition students environmental factors
 3. Maintaining staff records
 4. Finding relevant details of teachers when required
 5. Students' duty chart
 - ⑩ Classroom environment
 - ⑩ Toilets
 - ⑩ Surrounding