HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN I

BÀI GIẢNG

TIN HỌC QUẢN LÝ

NGƯỜI VIẾT: Ths. TRỊNH THỊ VÂN ANH

HÀ NỘI - 2013

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN VỀ BÀI TOÁN QUẢN LÝ VÀ XÂY DỰNG CSDL CHO BÀI TOÁN QUẢN LÝ

1.1. Tổng quan về bài toán quản lý

Công nghệ thông tin có thể hỗ trợ doanh nghiệp cải thiện hiệu quả và hiệu suất của các qui trình nghiệp vụ kinh doanh, quản trị ra quyết định, cộng tác nhóm làm việc, qua đó tăng cường vị thế cạnh tranh của doanh nghiệp trong một môi trường thay đổi nhanh. Tin học hóa công tác quản lý của các đơn vị kinh tế, hành chính...(tin học quản lý) đang là lĩnh vực quan trọng nhất của ứng dụng tin học. Xây dựng và phát triển hệ thống thông tin kinh tế và quản lý hiện đại là nội dung chủ yếu của ứng dụng tin học trong việc tự động hóa từng phần hoặc toàn bộ các quy trình nghiệp vụ, quản lý trong các tổ chức kinh tế.

1.1.1. Tập các bài toán thực tế

Hệ thống là một tập hợp gồm nhiều phần tử tương tác, có các mối quan hệ ràng buộc lẫn nhau và cùng hoạt động hướng tới một mục tiêu chung thông qua chấp thuận các đầu vào, biến đổi có tổ chức để tạo kết quả đầu ra.

Ví dụ:

- Hệ thống điều khiển giao thông
- Hệ thống mạng máy tính
- ...

Các thành phần của hệ thống:

- Nhập vào (Input): Nắm bắt và tập hợp các yếu tố để đưa vào hệthống để xử lý
- Xử lý (Processing): Bước biến đổi nhằm chuyển các yếu tố đưa vào sang các dạng cần thiết
- Kết xuất (Output): Chuyển các yếu tố được tạo ra từ quá trình xử lý thành các kết quả cuối cùng

Hệ thống thông tin: là một tập hợp các phần cứng, phần mềm, hệ mạng truyền thông được xây dựng và sử dụng để thu thập, tạo, tái tạo, phân phối và chia sẻ dữ liệu, thông tin và tri thức nhằm phục vụ các mục tiêu của tổ chức.

Hệ thống thông tin quản lý: Một hệ thống tích hợp "Người - Máy" tạo ra các thông tin giúp con người trong sản xuất, quản lý và ra quyết định là hệ thông tin quản lý. Hệ thông tin quản lý sử dụng các thiết bị tin học, các phần mềm, CSDL, các thủ tục thủ công, các mô hình để phân tích, lập kế hoạch quản lý và ra quyết định.

Những tác động của CNTT tới một trong các ngành sau:

+ Dịch vụ tài chính
+ Chăm sóc sức khỏe
+ Sản xuất
+ Dịch vụ giải trí nghe nhìn
+ Giáo dục
+ Bán lẻ
+ Du lịch và khách sạn

Một số công ty ứng dụng CNTT thành công trên thế giới:

+ Boeing Airplane Company

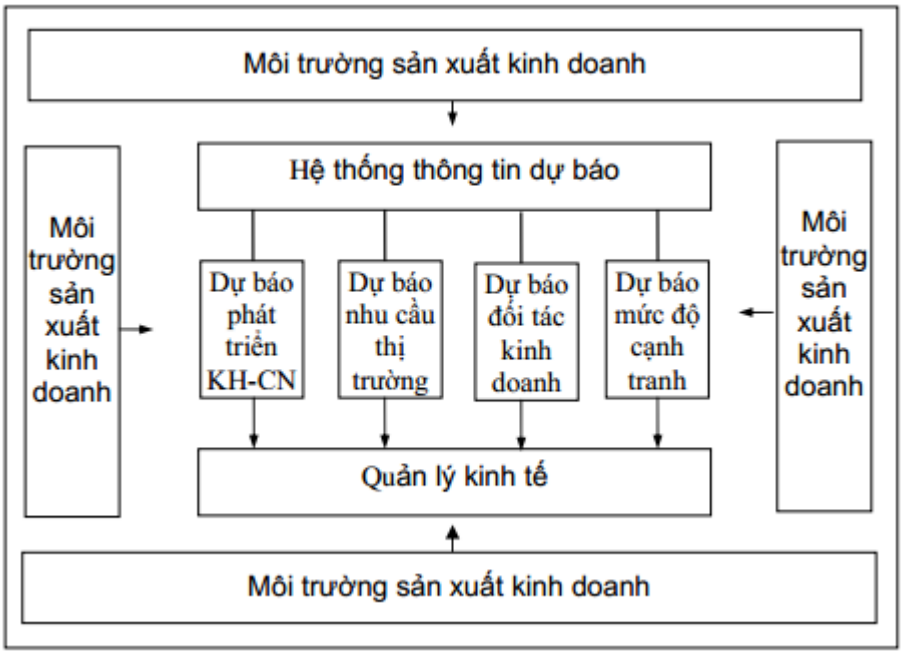
+ Wal-Mart Stores
+ Bissett Nursery Corp.
+ Federal Express
+ Charles Schwab
+ USAA
+ L.L. Bean
+ Progressive Corp.

1.1.2. Sơ đồ khối của bài toán cụ thể

Để định hướng ra các hệ thống thông tin quản lý người ta dựa vào định hướng hoạt động của hệ thống thông tin và tổng thể các bài toán quản lý mà hệ thống giải quyết. Theo cách này thì có thể chia các hệthống thông tin thành một số dạng sau:

- Hệ thống thông tin dự báo
- Hệ thống thông tin khoa học
- Hệ thống thông tin kế hoạch
- Hệ thống thông tin thực hiện

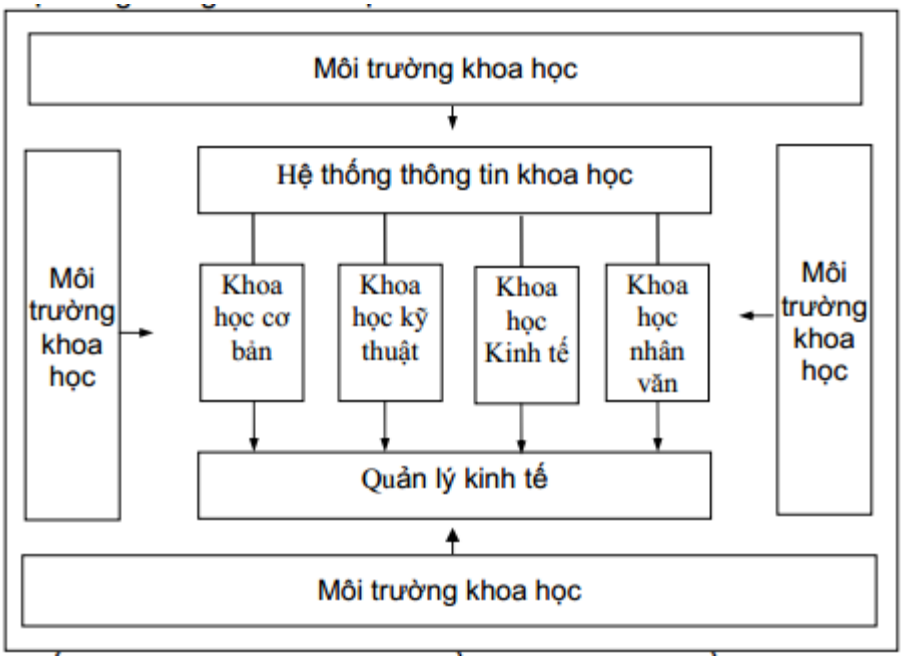
Hệ thống thông tin dự báo:



Hình 1.1: Mô hình hệ thống thông tin dự báo

Hệ thống thông tin dự báo bao gồm DB dài hạn, DB trung hạn và DB ngắn hạn về các vấn đề liên quan đến doanh nghiệp, như DB các tiến độ KH-CN, dự báo qui mô sản xuất, dự báo về nhu cầu của thị trường, về mức độ cạnh tranh trên thị trường,… Hệ thống thông tin dự báo càng quan trọng trong hoạt động kinh tế thị trường.

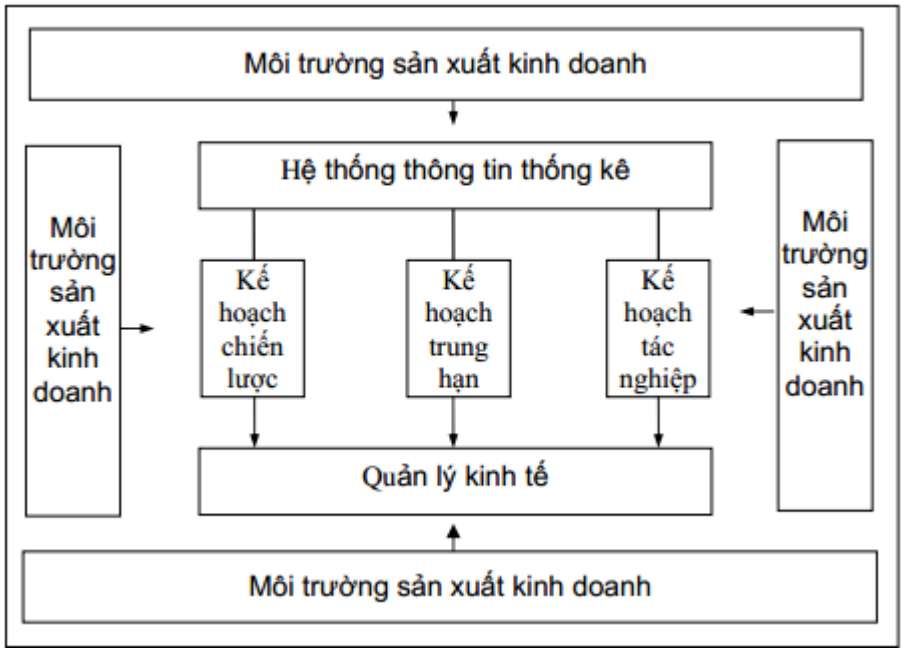
Hệ thống thông tin khoa học:



Hình 1.2: Mô hình hệ thống thông tin khoa học

Hệ thống thông tin khoa học bao gồm các thông tin về KHCB, KHCN, KHKT và KHTN. Từ môi trường KH rộng lớn hệ thống thông tin khoa học thu thập các thông tin liên quan đến sản xuất - kinh doanh để đáp ứng yêu cầu quản lý của doanh nghiệp.

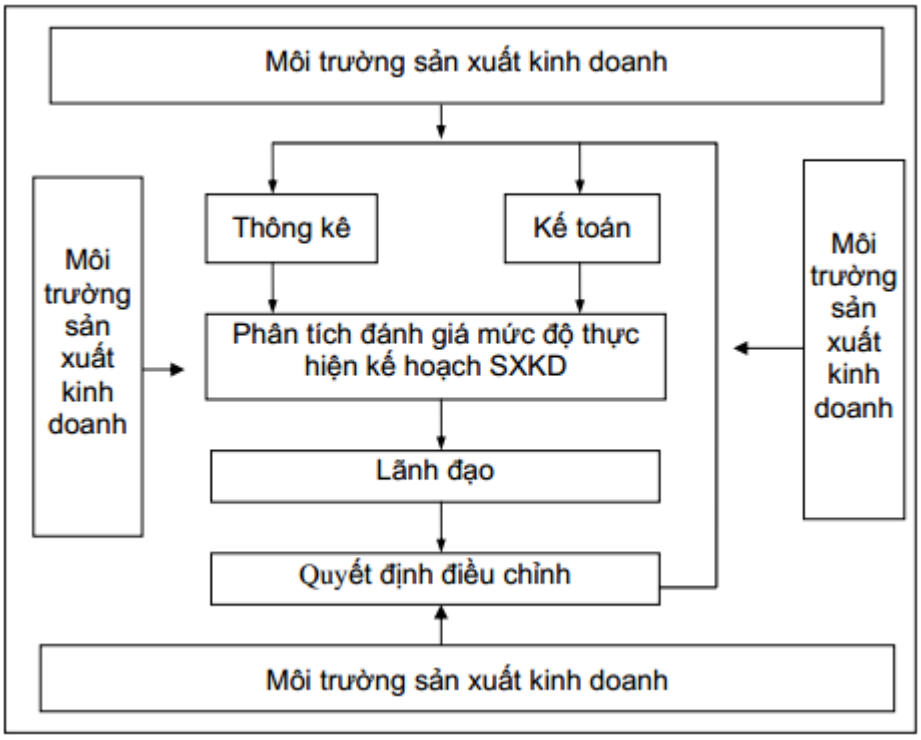
Hệ thống thông tin kế hoạch:



Hình 1.3: Mô hình hệ thống thông tin kế hoạch

Hệ thống thông tin kế hoạch bao gồm toàn bộ các thông tin về công tác kế hoạch hoá của doanh nghiệp. Các kế hoạch được đề cập đến ở 3 mức độ: kế hoạch hóa chiến lược, kế hoạch hóa trung hạn, kế hoạch hóa cơ động. HTTTKH bao quát tất cả các lĩnh vực hoạt động của doanh nghiệp gồm cả trong lĩnh vực sản xuất và quản lý.

Hệ thống thông tin thực hiện:



Hình 1.4: Mô hình hệ thống thông tin thực hiện

Hệ thống thông tin thực hiện sử dụng các công cụ thống kê và kế toán để kiểm tra, đánh giá, phân tích các quá trình thực hiện kế hoạch theo thời gian. Trên cơ sở các số liệu của HTTT thực hiện, mà cán bộ lãnh đạo có thể ra các quyết định điều chỉnh.

Như vậy, phương pháp phân loại thông tin theo nội dung giúp chúng ta định hướng rõ mục đích của các dòng thông tin trong hệ thống quản lý. Trên cơ sở đó tiến hành phát triển hoàn thiện một nội dung nào đó trong hệ thống quản lý.

1.1.3. Phân tích và thiết kế bài toán cụ thể

Giả sử với bài toán xây dựng hệ thống ứng dụng cẩm nang hỗ trợ người dùng trong ẩm thực và nấu ăn.

Đặc tả chức năng:

Chức năng tìm kiếm món ăn: Cho phép tìm kiếm món ăn theo ba miền bắc, trung, nam. Mỗi miền gồm mười hai loại món ăn khác nhau. Với mỗi loại món ăn cho phép người dùng xem một danh sách món ăn đặc sắc thuộc loại đó. Cho phép người dùng xem chi tiết về món ăn: ảnh món ăn, các loại gia vị cùng số lượng (nếu có), tác dụng gia vị, cách chế biến món ăn. Cho phép người dùng đánh dấu món ăn vào danh sách món ăn yêu thích.

Chức năng xem video nấu ăn: Cho phép người dùng xem danh sách các video nấu ăn của các đầu bếp nổi tiếng. Người dùng chọn một video về món ăn của đầu bếp mà người dùng thích để xem hướng dẫn chế biến.

Chức năng tim kiếm nhà hàng: Cho phép người dùng tim kiếm nhà hàng theo từng khu vực Huế, Hà Nội, Sài Gòn. Với mỗi khu vực hiển thị danh sách các nhà hàng thuộc khu vực đó. Cho phép người dùng xem thông tin chi tiết về nhà hàng: ảnh nhà hàng, địa chỉ, điện thoại liên lạc, giờ mở cửa đóng cửa, website.

Chức năng xem quảng cáo nhà hàng: Cho phép người dùng xem thông tin về các chương trình khuyến mại của các nhà hàng, người dùng liên hệ với nhà hàng thông qua số điện thoại để biết thêm thông tin.

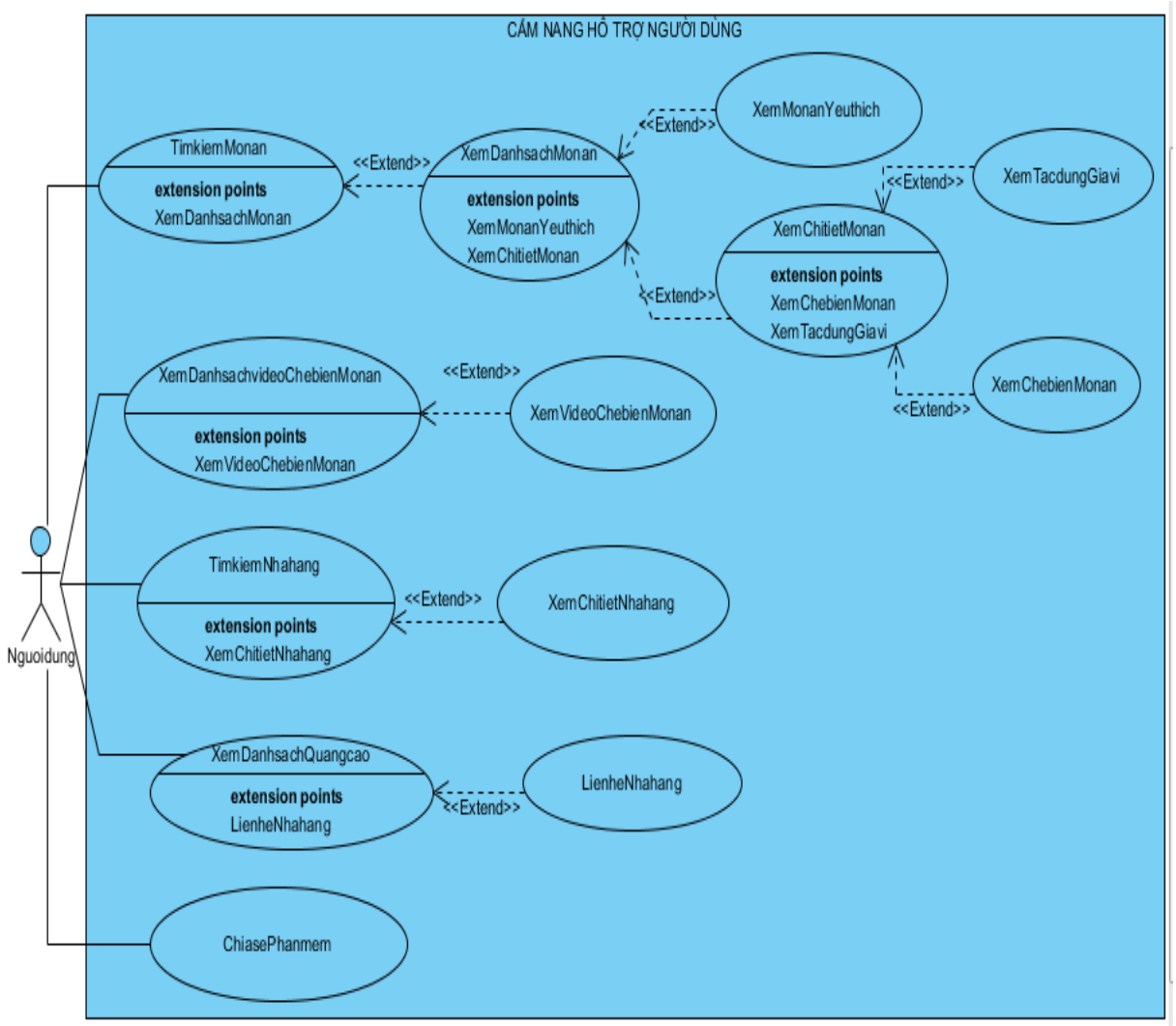
Chức năng chia sẻ phần mềm: Cho phép người dùng chia sẻ thông tin về phần mềm qua sms, facebook, gmail bao gồm: tên phần mềm, mô tả phần mềm, đường dẫn.

Phân tích hệ thống:

Biểu đồ Usecase tổng quát của hệ thống

Bảng 1.1: Danh sách các usecase

TT	Tên usecase	Ý nghĩa
1	TimkiemMonan (Tìm kiếm món ăn)	Chức năng này cho phép người dùng xem danh sách các loại món ăn ba miền Bắc, Trung, Nam.
2	XemDanhsachMonan (Xem danh sách món ăn)	Chức năng này cho phép người dùng xem danh sách các món ăn của một loại món ăn.
3	XemChitietMonan (Xem món ăn)	Chức năng này cho phép người dùng xem chi tiết món ăn.
4	XemDanhsachMonanYeuthich (Xem danh sách món ăn yêu thích)	Chức năng này cho phép người dùng xem danh sách các món ăn yêu thích được người dùng đánh dấu.
5	XemTacdungGiavi (Xem tác dụng gia vị)	Chức năng này cho phép người dùng xem tác dụng của gia vị trong món ăn đã chọn.
6	XemChebienMonan(Xem chế biến món ăn)	Chức năng này cho phép người dùng xem các bước chế biến món ăn đã chọn.
7	TimkiemNhahang (Tìm kiếm nhà hàng)	Chức năng này cho phép người dùng tìm kiếm các nhà hàng ở Hà Nội, Huế, Sài Gòn.
8	XemChitietNhahang (Xem chi tiết nhà hàng)	Chức năng này cho phép người dùng xem chi tiết các thông tin về nhà hàng (Hình ảnh, địa chỉ, số điện thoại liên hệ, website nhà hàng, giờ mở cửa, giờ đóng cửa, các đánh giá của người dùng đối với nhà hàng…)
9	XemDanhsachQuangcao (Xem danh sách quảng cáo)	Chức năng này cho phép người dùng xem danh sách quảng cáo của các nhà hàng.
10	Lienhe (Liên hệ)	Chức năng này cho phép người dùng liên hệ với nhà hàng.
11	XemDanhsachvideoChebienMonan(Xem danh sách video chế biến món ăn)	Chức năng này cho phép người dùng xem danh sách các video hướng dẫn nấu ăn của các món.
12	XemVideoChebienMonan (Xem video chế biến món ăn)	Chức năng này cho phép người dùng xem video hướng dẫn chế biến món ăn người dùng đã chọn.
13	ChiasePhanmem (Chia sẻ phần mềm)	Chức năng này cho phép người dùng chia sẻ phần mềm bạn bè qua sms, gmail, facebook.



Hình 1.5: Sơ đồ usecase tổng quát của hệ thống cẩm nang hỗ trợ người dùng

Biểu đồ lớp phân tích

Lớp biên:

- GDDanhsachLoaiMonan (Giao diện danh sách loại món ăn)
- GDDanhsachMonan (Giao diện danh sách món ăn)
- GDChitietMonan (Giao diện chi tiết món ăn)
- GDChitietMonan (Giao diện chi tiết món ăn)
- GDTacdungGiavi (Giao diện tác dụng gia vị)
- GDDanhsachNhahang (Giao diện danh sách nhà hàng)
- GDThongtinChitietNhahang (Giao diện thông tin chi tiết nhà hàng)
- GDDanhsachvideoChebienMonan (Giao diện danh sách video chế biến món ăn)

Lớp thực thể:

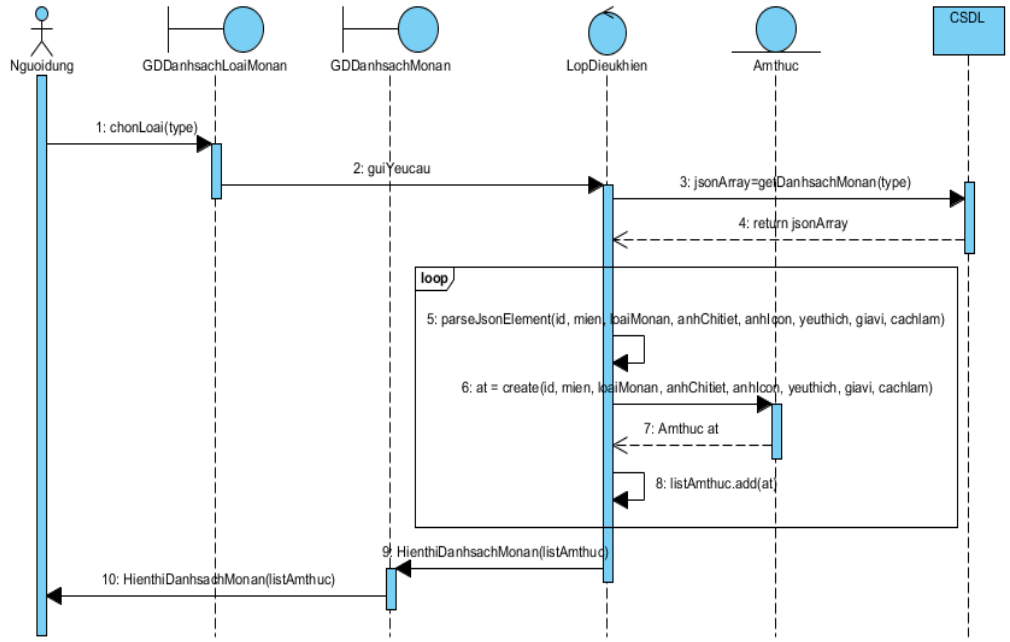
- Amthuc (Ẩm thực)
- Quangcao (Quảng cáo)
- Diadiem (Địa điểm)
- ChitietDiadiem (Chi tiết địa điểm)
- Giavi (Gia vị)
- Nhahang (Nhà hàng)

Lớp điều khiển:

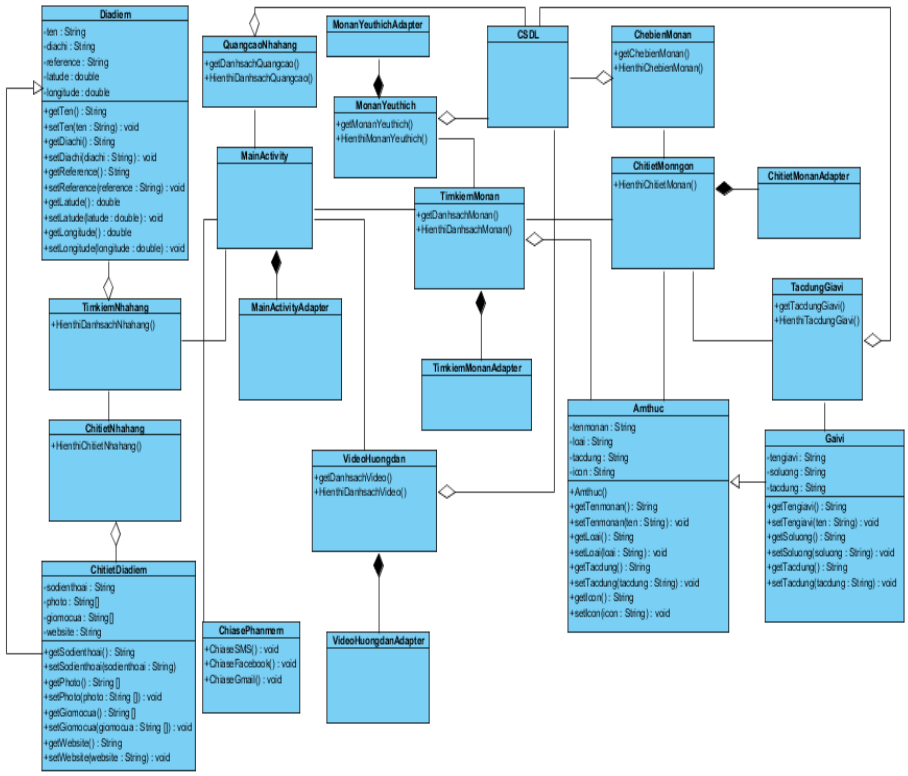
- LopDieukhien (Lớp điều khiển)

Thiết kế hệ thống:

Thiết kế cho từng chức năng.



Hình 1.6: Biểu đồ tuần tự cho chức năng TimkiemMonngon



Hình 1.7: Sơ đồ lớp chi tiết

1.2. Các khái niệm về CSDL

1.2.1. Các khái niệm chung

Hệ quản trị cơ sở dữ liệu (Database Management System - DBMS): Là một hệ thống phần mềm cho phép tạo lập cơ sở dữ liệu và điều khiển mọi truy nhập đối với cơ sở dữ liệu

đó. Trên thị trường phần mềm hiện nay ở Việt Nam đã xuất hiện khá nhiều phần mềm hệ quản trị cơ sở dữ liệu như: Microsoft Access, Foxpro, DB2, SQL Server, Oracle,.v.v…

Hệ quản trị cơ sở dữ liệu quan hệ (Relation Database Management System - RDBMS) là một hệ quản trị cơ sở dữ liệu theo mô hình quan hệ.

Một hệ quản trị cơ sở dữ liệu thường cung cấp hai kiểu ngôn ngữ khác nhau đó là: ngôn ngữ mô tả sơ đồ cơ sở dữ liệu và ngôn ngữ biểu diễn các truy vấn và các cập nhật cơ sở dữ liệu.

- Ngôn ngữ định nghĩa dữ liệu (Data Definition Language - DDL)

+ Một sơ đồ CSDL đặc tả bởi một tập các định nghĩa được biểu diễn bởi một ngôn ngữ đặc biệt được gọi là ngôn ngữ định nghĩa dữ liệu. Kết quả của việc dịch các ngôn ngữ này là một tập các bảng được lưu trữ trong một tệp đặc biệt được gọi là từ điển dữ liệu hay thư mục dữ liệu.

+ Một từ điển dữ liệu là một tệp chứa các siêu dữ liệu có nghĩa là các dữ liệu về dữ liệu. Tệp này được tra cứu trước khi dữ liệu thực sự được đọc hay được sửa đổi trong hệ CSDL.

+ Cấu trúc và các phương pháp truy nhập được sử dụng bởi hệ CSDL được đặc tả bởi một tập các định nghĩa trong một kiểu đặc biệt của DDL là ngôn ngữ định nghĩa và lưu trữ dữ liệu.

- Ngôn ngữ thao tác dữ liệu (Data Manipulation Language - DML):

+ Các yêu cầu về thao tác dữ liệu bao gồm:

• Tìm kiếm thông tin được lưu trữ trong CSDL.

• Thêm thông tin mới vào CSDL.

• Xoá thông tin từ CSDL.

• Thay đổi thông tin được lưu trữ trong CSDL.

+ Một ngôn ngữ thao tác dữ liệu (DML) là một ngôn ngữ cho phép người sử dụng truy nhập hay thao tác dữ liệu được tổ chức bởi mô hình dữ liệu thích hợp. Có hai kiểu ngôn ngữ thao tác dữ liệu cơ bản:

• Các DML thủ tục đòi hỏi người sử dụng phải đặc tả dữ liệu nào cần tìm kiếm và tìm kiếm những dữ liệu này như thế nào.

• Các DML phi thủ tục đòi hỏi người sử dụng đặc tả dữ liệu nào cần tìm kiếm mà không phải đặc tả tìm kiếm những dữ liệu này như thế nào.

1.2.2. Các phép toán của bài toán CSDL

Ngôn ngữ SQL (Structured Query Language) là ngôn ngữ truy vấn có cấu trúc, dùng để thao tác với dữ liệu trong cơ sở dữ liệu cũng như tạo và thay đổi cấu trúc của các cơ sở dữ liệu. Trong chương này ta sẽ trình bày một số câu lệnh SQL cơ bản.

Lệnh CREATE dùng để tạo các đối tượng cơ sở dữ liệu như các bảng, các view, các tệp chỉ số.v.v…

- Cú pháp:

+ CREATE TABLE <Tên bảng>(<Danh sách: Tên_cột

Kiểu_cột> <Điều_kiện_kiểm_soát_dl >)

+ CREATE VIEW <Tên View>(<Danh sách: Tên_cột

Kiểu_cột> <Điều_kiện_kiểm_soát_dl >) AS Q; với Q là một khối câu lệnh SELECT định nghĩa khung nhìn (view).

+ CREATE [UNIQUE] INDEX <tên chỉsố> ON <Ten bảng>(Tên cột [ASC|DESC])

- Một số kiểu dữ liệu: Integer - sốnguyên; float - dấu phảy động; char - ký tự, datetime - ngày tháng, boolean,…

Lệnh ALTER: dùng để thay đổi cấu trúc lược đồ của các đối tượng CSDL.

- Cú pháp:

+ ALTER TABLE <Tên bảng> <Thực hiện các lệnh trên cột>

Các lệnh trên cột có thể là:

• Xóa một cột: Delete <tên cột>

• Thêm một cột: Add <Tên cột>

• Thay đổi tên cột: Change column <Tên cột>To<Tên cột>

• Xóa khóa chính: Drop PRIMARY KEY

• Xóa khóa ngoại: Drop FOREIGN KEY

• Thiết lập khóa chính: PRIMARY KEY (Tên cột)

• Thiết lập khóa ngoại:

FOREIGN KEY (Tên cột) REFERENCES TO <tên bảng ngoài>

+ ALTER VIEW <Tên View>(<Danh sách: Tên_cột Kiểu_cột>

<Điều_kiện_kiểm_soát_dl >) AS Q; với Q là một khối câu lệnh

SELECT định nghĩa khung nhìn (view).

Lệnh DROP: dùng để xóa các đối tượng cơ sở dữ liệu như Table, View,

Index, .v.v…

- Cú pháp:

DROP TABLE <Tên bảng>

DROP VIEW <Tên view>

DROP INDEX <Tên index>

Lệnh INSERT INTO: dùng để chèn một hàng hoặc một số hàng cho bảng.

- Cú pháp:

+ INSERT INTO <Tên bảng> (Danh sách các cột) VALUES

(Danh sách các giá trị) hoặc

+ INSERT INTO <Tên bảng> (Danh sách các cột) (Các câu hỏi con);

Lệnh UPDATE: dùng để sửa đổi dữ liệu.

- Cú pháp:

UPDATE <Tên bảng>

SET <Tên_cột_1=Biểu_thức_1, Tên_cột_2=Biểu_thức_2,… >[WHERE <điều kiện>]

Lệnh DELETE FROM: Xoá một số hàng trong bảng.

- Cú pháp:

DELETE FROM <Tên bảng> WHERE <Điều kiện>

Khối câu lệnh phổ dụng: SELECT - FROM – WHERE. Ta có thể sử dụng theo cú pháp chung như sau:

SELECT [*| DISTINCT] <Danh sách các cột [AS <Bí danh>]>

FROM <Danh sách Tên bảng/Tên View>

[WHERE <Biểu thức điều kiện>]

[GROUP BY <Danh sách cột>]

[HAVING <Điều kiện>]

[ORDER BY <Tên cột/ Số thứ tự cột/Biểu thức> [ASC/DESC]]

CHƯƠNG 2: NHẬP MÔN VISUAL BASIC.NET

2.1 Bắt đầu với VB.NET

Ngôn ngữ **BASIC** (Beginner's All Purpose Symbolic Instruction Code) đã có từ năm 1964, BASIC rất dễ học và dễ dùng. Trong vòng 15 năm đầu, có rất nhiều chuyên gia Tin Học và công ty tạo các chương trình thông dịch (Interpreters) và biên dịch (Compilers) cho ngôn ngữ làm BASIC trở nên rất phổ thông.

Năm 1975, Microsft tung ra thị trường sản phẩm đầu tay Microsoft BASIC và tiếp đó Quick BASIC (còn gọi là **QBASIC**) thành công rực rỡ.

Quick BASIC phát triển trong nền Windows nhưng vẫn khó khăn khi tạo giao diện kiểu Windows. Sau đó nhiều năm, Microsoft bắt đầu tung ra một sản phẩm mới cho phép ta kết hợp ngôn ngữ dễ học BASIC và môi trường phát triển lập trình với giao diện bằng hình ảnh (Graphic User Interface - **GUI**) trong Windows. Đó là Visual Basic Version 1.0.

Sự chào đời của Visual Basic Version 1.0 vào năm 1991 thật sự thay đổi bộ mặt lập trình trong Công Nghệ Tin Học.

Trước đó, ta không có một giao diện bằng hình ảnh (GUI) với một **IDE (Integrated Development Environment)** giúp các chuyên gia lập trình tập trung công sức và thì giờ vào các khó khăn liên hệ đến doanh nghiệp của mình. Mỗi người phải tự thiết kế giao diện qua thư viện có sẵn **Windows API (Application Programming Interface)** trong nền Windows. Điều này tạo ra những trở ngại không cần thiết làm phức tạp việc lập trình.

Visual Basic giúp ta bỏ qua những hệ lụy đó, chuyên gia lập trình có thể tự **vẽ** cho mình giao diện cần thiết trong ứng dụng (application) một cách dễ dàng và như vậy, tập trung nổ lực giải đáp các vần đề cần giải quyết trong doanh nghiệp hay kỹ thuật.

Ngoài ra, còn nhiều công ty phụ phát triển thêm các khuôn mẫu (modules), công cụ (tools, controls) hay ứng dụng (application) phụ giúp dưới hình thức **VBX** cộng thêm vào giao diện chính càng lúc càng thêm phong phú.

Khi Visual Basic phiên bản 3.0 được giới thiệu, thế giới lập trình lại thay đổi lần nữa. Kỳ này, ta có thể thiết kế các ứng dụng (application) liên hệ đến **Cơ Sở Dữ Liệu (Database)** trực tiếp tác động (interact) đến người dùng qua **DAO (Data Access Object)**. Ứng dụng này thường gọi là **ứng dụng tiền diện (front-end application) hay trực diện.**

Phiên bản 4.0 và 5.0 mở rộng khả năng VB nhắm đến Hệ Điều Hành Windows 95.

Phiên bản 6.0 cung ứng một phương pháp mới nối với Cơ Sở Dữ Liệu (Database) qua sự kết hợp của **ADO (Active Data Object)**. ADO còn giúp các chuyên gia phát triển mạng nối với Cơ Sở Dữ Liệu (Database) khi dùng Active Server Pages (ASP).

Tuy nhiên, VB phiên bản 6.0 (VB6) không cung ứng tất cả các đặc trưng của kiểu mẫu **ngôn ngữ lập trình khuynh hướng đối tượng** (**O**bject **O**riented **L**anguage - **OOL**) như các ngôn ngữ C++, Java.

Thay vì cải thiện hay vá víu thêm thắt vào VB phiên bản 6.0, Microsoft đã xoá bỏ tất cả làm lại từ đầu các ngôn ngữ lập trình mới theo kiểu OOL rất hùng mạnh cho khuôn nền .NET Framework. Đó là các ngôn ngữ lập trình **Visual Basic.NET và C# (gọi là C Sharp)**. Sau đó, nhiều ngôn ngữ lập trình khác cũng thay đổi theo ví dụ như smalltalk.NET, COBOL.NET, ... làm Công Nghệ Tin Học trở nên phong phú hơn, đa dạng hơn.

Tất cả những thay đổi này nhằm đáp ứng kịp thời sự đòi hỏi và nhu cầu phát triển cấp bách trong kỹ nghệ hiện nay.

Visual Basic.NET (VB.NET) là ngôn ngữ lập trình khuynh hướng đối tượng **(Object Oriented Programming Language)** do Microsoft thiết kế lại từ con số không. Visual Basic.NET (VB.NET) không kế thừa VB6 hay bổ sung, phát triển từ VB6 mà là một ngôn ngữ lập trình hoàn toàn mới trên nền Microsoft's .NET Framework. Do đó, nó cũng không phải là VB phiên bản 7. Thật sự, đây là ngôn ngữ lập trình mới và rất lợi hại, không những lập nền tảng vững chắc theo kiểu mẫu đối tượng như các ngôn ngữ lập trình hùng mạnh khác đã vang danh C++, Java mà còn dễ học, dễ phát triển và còn tạo mọi cơ hội hoàn hảo để giúp ta giải đáp những vấn đề khúc mắc khi lập trình. Hơn nữa, dù không khó khăn gì khi cần tham khảo, học hỏi hay đào sâu những gì xảy ra bên trong ... hậu trường OS, Visual Basic.NET (VB.NET) giúp ta đối phó với các phức tạp khi lập trình trên nền Windows và do đó, ta chỉ tập trung công sức vào các vấn đề liên quan đến dự án, công việc hay doanh nghiệp mà thôi.

Trong khóa học này, các bạn sẽ bắt đầu làm quen với kiểu lập trình dùng Visual Basic.NET (VB.NET) và dĩ nhiên, các khái niệm và thành phần cơ bản của .NET Framework.

Nếu ta để ý tên của Visual Basic**.NET** (VB.NET), ta thấy ngay ngôn ngữ lập trình này chuyên tạo ứng dụng (application) dùng trong mạng, liên mạng hay trong Internet. Do đó, ta sẽ tập trung vào việc lập trình các ứng dụng (applications) trên nền Windows và đó cũng là mục tiêu chính yếu khi học Visual Basic.NET cơ bản.

2.2 Cài đặt Microsoft Visual Studio.NET

Bộ Microsoft Visual Studio.NET bao gồm vừa mọi công cụ yểm trợ lập trình và ngôn ngữ lập trình .NET, tỷ như: **Visual Basic.NET (VB.NET), C# (C Sharp), Visual C++.NET và Visual J#.NET**

Tùy ý ta chọn loại ngôn ngữ lập trình nào thích hợp để cài vào máy vi tính. Không ai cấm ta cài đủ thứ vào máy nhưng dĩ nhiên cần phải có dư chỗ trong hard drive, Microsoft Visual Studio.NET sẽ tính toán và cho ta biết khả năng chứa như thế nào. Tuy nhiên, ta có thể chỉ chọn Visual Basic.NET (VB.NET) và các ứng dụng (application) liên hệ trước, nếu cần học thêm về C# hay Visual C++.NET, ta có thể cài sau cũng được vì nếu cài toàn bộ, ta sẽ cần khoãng trên dưới 1.5 GBytes trong hard drive.

Microsoft Visual Studio.NET có nhiều phiên bản khác nhau. Dưới đây, ta tạm dùng phiên bản **Enterprise Architecture 2003** làm thí dụ điển hình. Tùy theo phiên bản ta có, những bước cài đặt sẽ khác nhau 1 chút nhưng trên nguyên tắc, ta phải cài đầy đủ môi trường .NET yểm trợ lập trình trước khi cài Microsoft Visual Studio.NET, tỷ như:

Microsoft .NET Framework

Microsoft FrontPage Web Extensions Client

Microsoft Access trong bộ MS Office Professional

Microsoft SQL Server - sẽ hướng dẫn cài và bố trí MS SQL Server cho khóa học trong bài Cơ Sở Dữ Liệu (Database)

và các ứng dụng (application) liên hệ (Microsoft Visual Studio.NET cho biết ta cần những gì) như hình trong bước thứ 3.

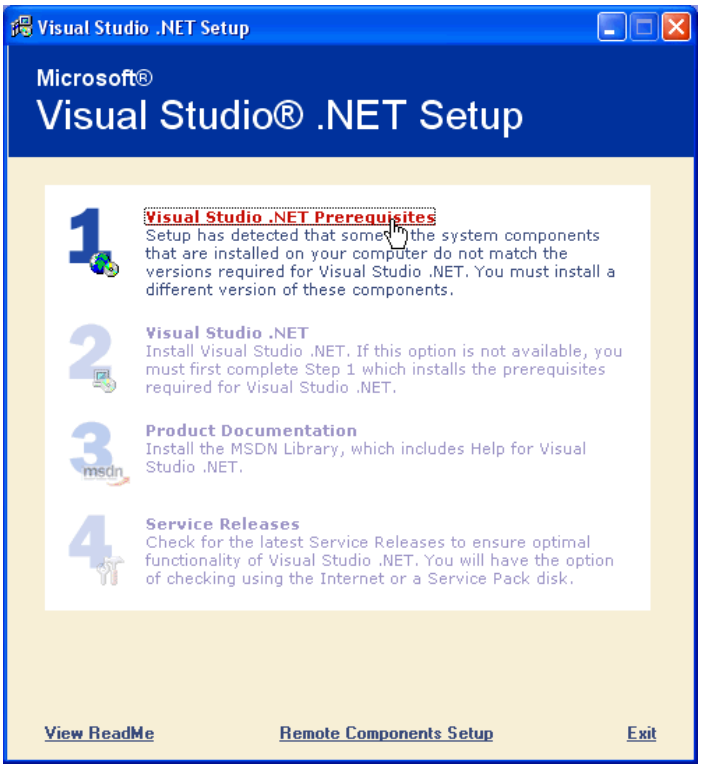
Bước 1:

Bắt đầu với dĩa 1 của bộ Microsoft Visual Studio.NET, dĩa này tự khởi động và hiển thị Windows hướng dẫn ta cài Microsoft Visual Studio.NET Setup. Nếu CD không tự khởi động được, ta cần chạy ứng dụng '**setup.exe**' trong vị trí gốc (root directory):

Chạy Windows Explorer, chọn dĩa cứng chứa Microsoft Visual Studio.NET Setup dĩa 1, nhấp đôi ứng dụng 'setup.exe' hay

Khởi động (Windows Start Menu) và chọn 'Run', gõ hàng chữ: 'e:\setup.exe' (nếu CD/DVD drive của ta là drive E).

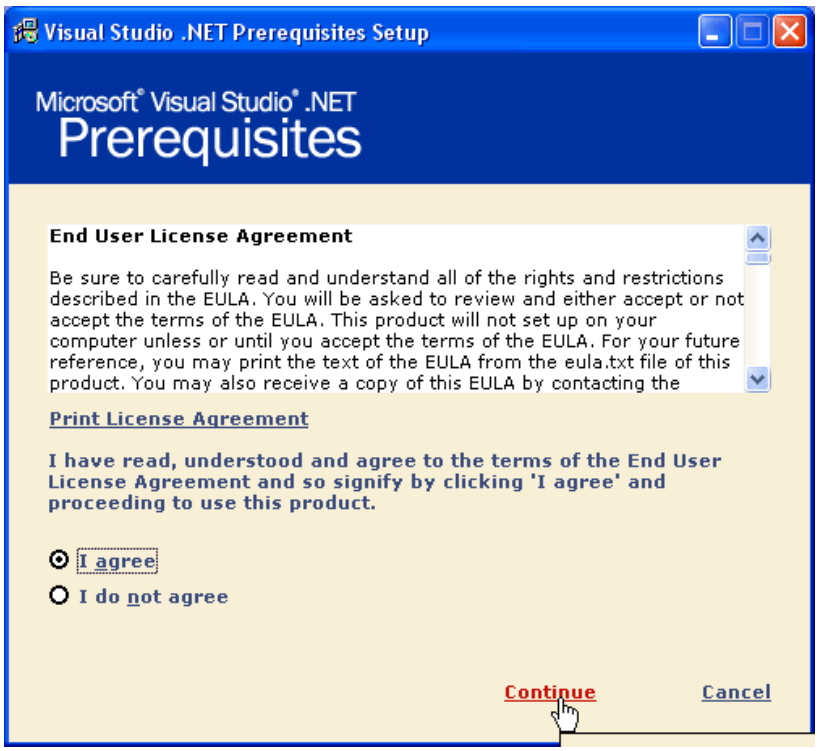
Microsoft Visual Studio.NET hiển thị 4 bước cài. Bước đầu tiên là chuẩn bị môi trường lập trình .NET với '**Visual Studio .NET Prerequisites**':



Hình 2.1: Màn hình bước 1

Bước 2:

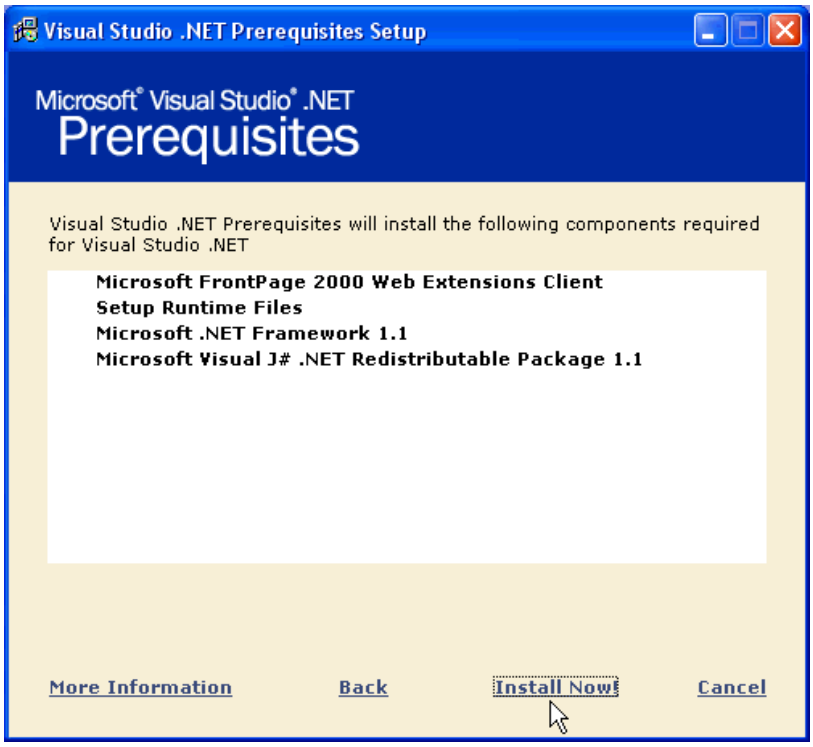
Nhập CD tên Microsoft Visual Studio.NET 2003 Prerequisites, chọn 'I agree' chấp nhận điều kiện dùng nhu liệu và nhấp Continue.



Hình 2.2: Màn hình bước 2

Bước 3:

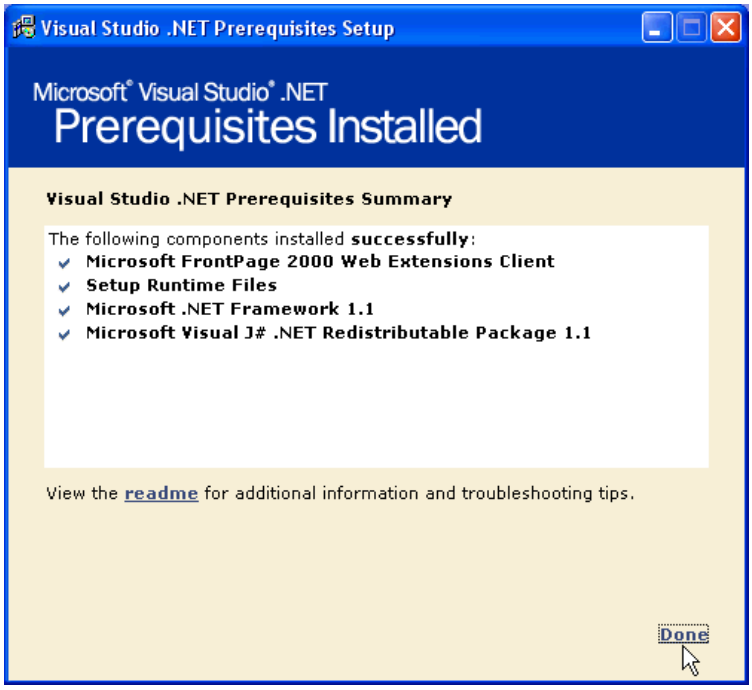
Nhấp Install Now! để cài các ứng dụng (application) liên hệ tạo môi trường .NET. Lưu ý ở đây, Microsoft Visual Studio.NET sẽ dò tìm những ứng dụng (application) cần thiết trong máy vi tính và tùy theo mỗi máy, bảng liệt kê ứng dụng có thể khác nhau. Thí dụ ở đây cho biết máy vi tính cần 4 ứng dụng phụ thuộc như hình sau:



Hình 2.3: Màn hình bước 3

Bước 4:

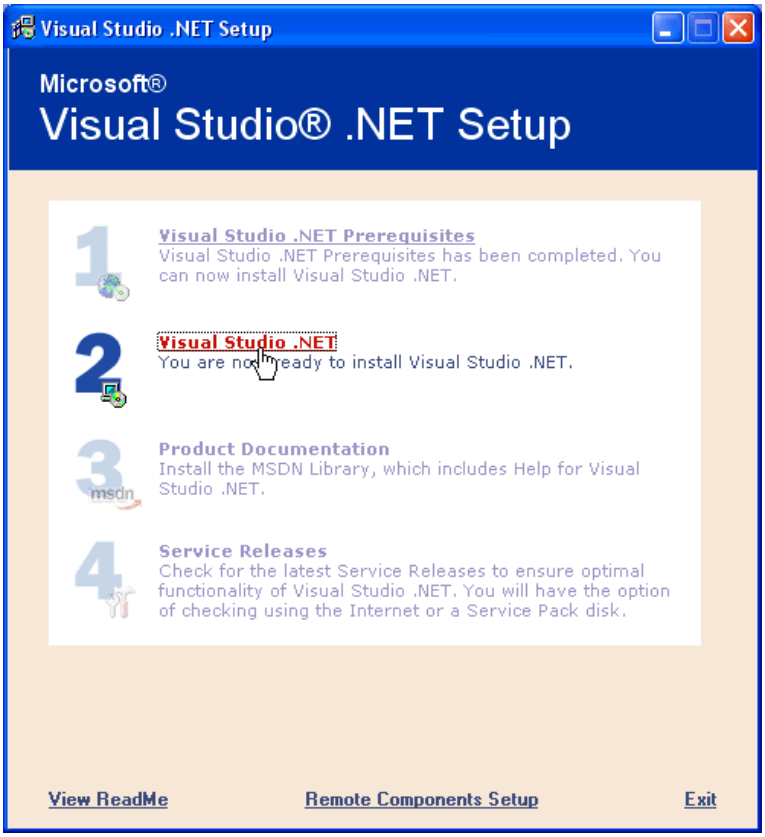
Chờ cho đến khi nào Microsoft Visual Studio.NET cài xong các ứng dụng phụ thuộc, nhấp nút Done.



Hình 2.4: Màn hình bước 4

Bước 5:

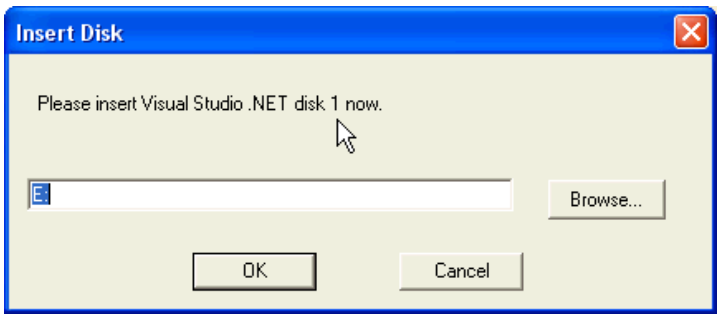
Tiếp tục chọn Visual Studio.NET



Hình 2.5: Màn hình bước 5

Bước 6:

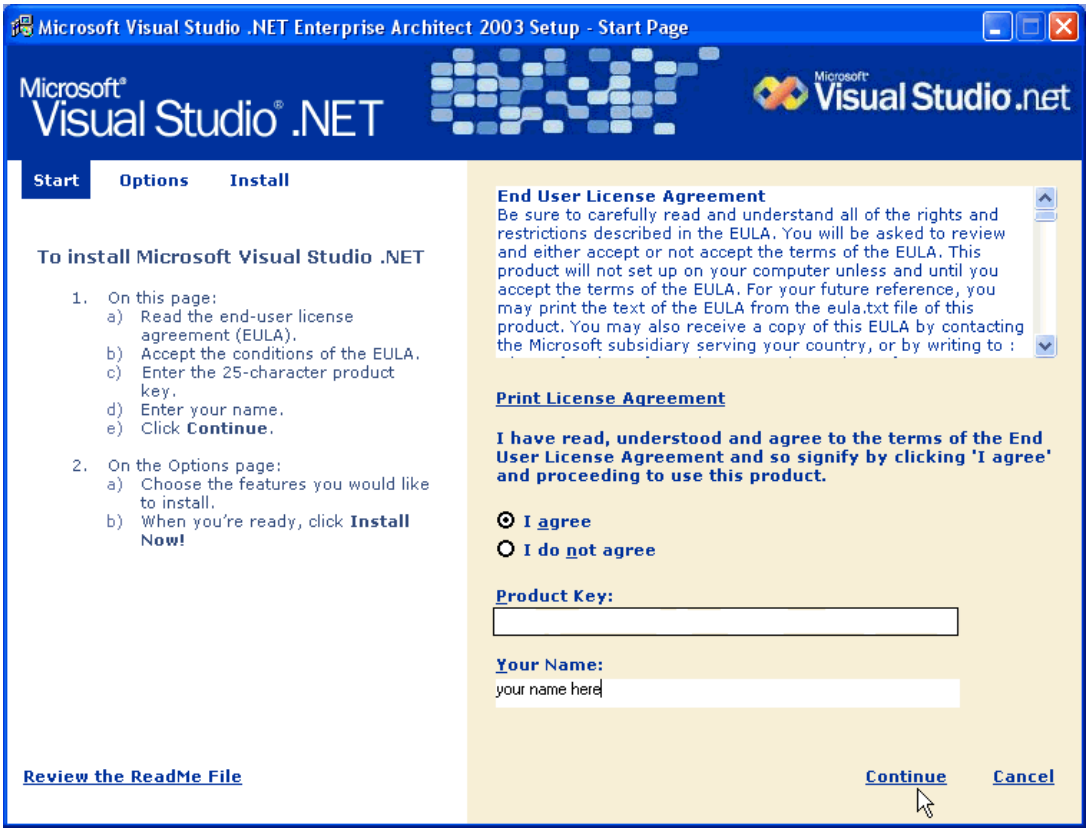
Nhập dĩa 1 vào máy và nhấp nút OK.



Hình 2.6: Màn hình bước 6

Bước 7:

Ta chọn 'I agree' và cung cấp Product Key trước khi nhấp nút Continue.

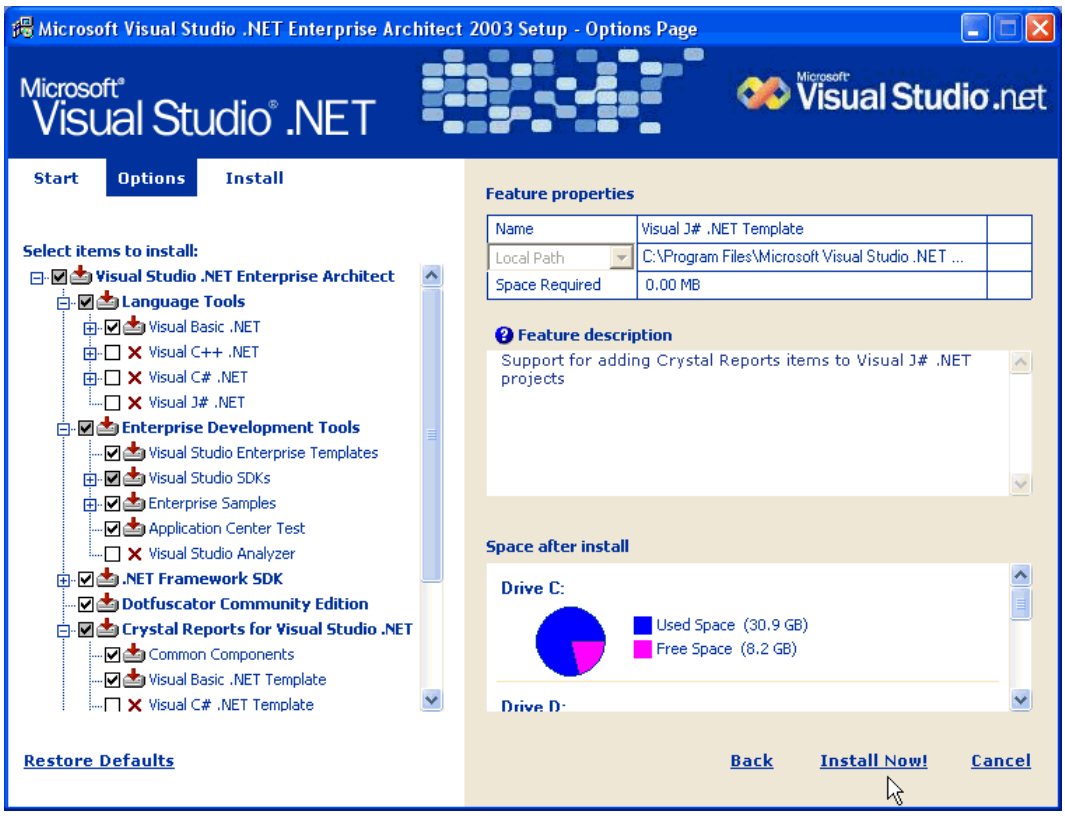


Hình 2.7: Màn hình bước 7

Bước 8:

Ta chỉ chọn những gì liên hệ đến Visual Basic.NET (VB.NET) cho khóa học Visual Basic.NET (VB.NET) Cơ Bản.

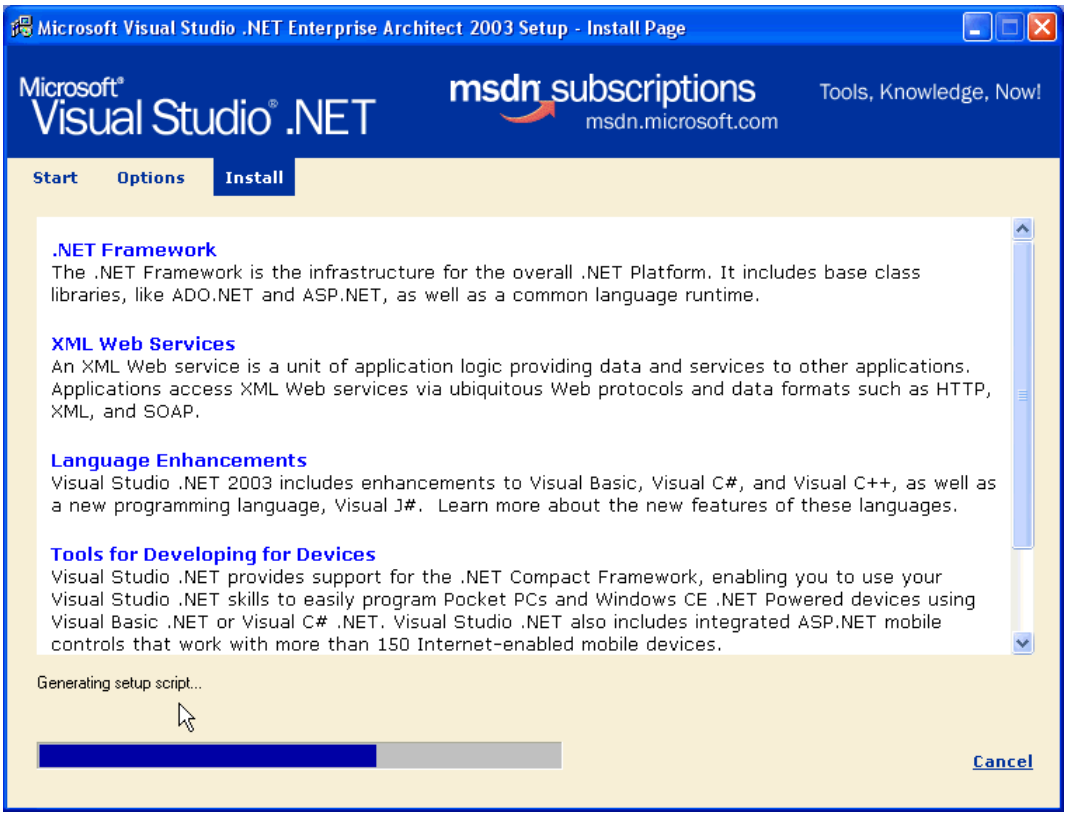
Xóa bỏ (uncheck) ngôn ngữ lập trình Visual C++.NET, Visual C#.NET, Visual J#.NET và các ứng dụng liên hệ, tỷ như: template, documetation, ...



Hình 2.8: Màn hình bước 8

Bước 9:

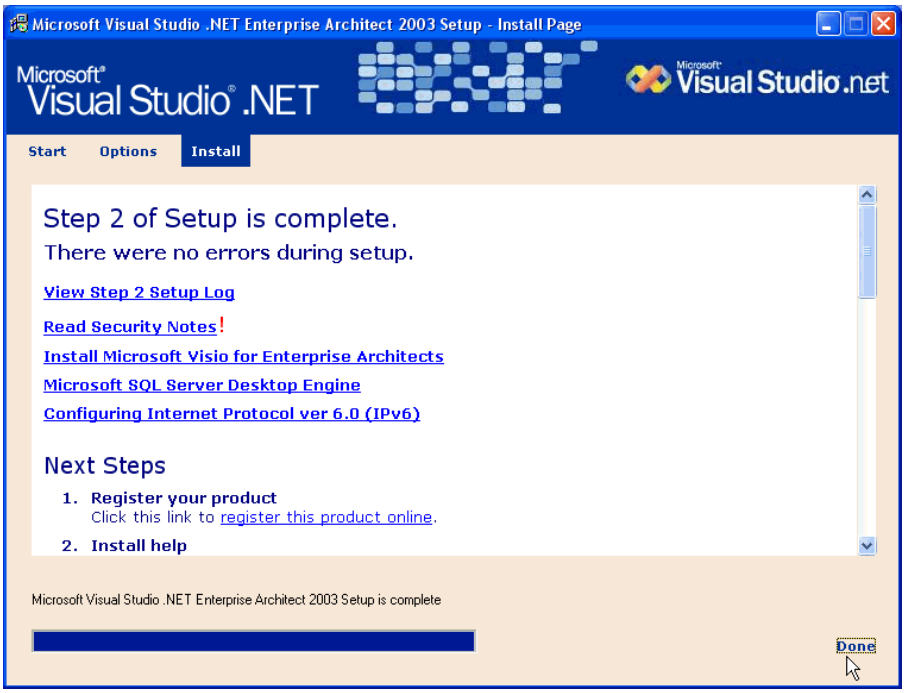
Nhấp Install Now. Microsoft Visual Studio.NET sẽ chạy ứng dụng cài và bố trí này khoãng trên dưới 1 tiếng đồng hồ tùy theo khả năng máy vi tính.



Hình 2.9: Màn hình bước 9

Bước 10:

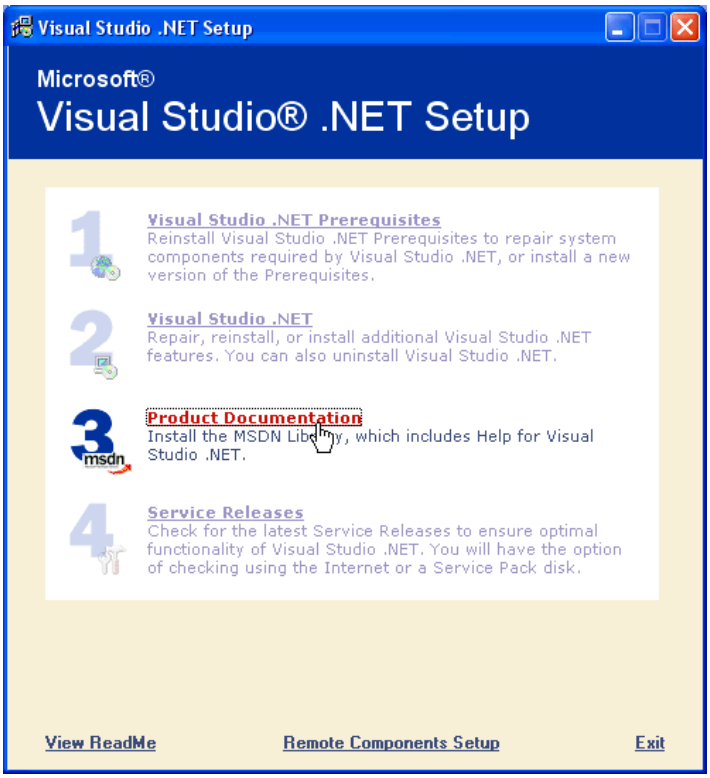
Nhấp **Done**. Microsoft Visual Studio.NET sẽ hiển thị Windows cài các thông tin phụ giúp lập trình và cả thư viện để ta tham khảo khi lập trình với Visual Basic.NET (VB.NET):

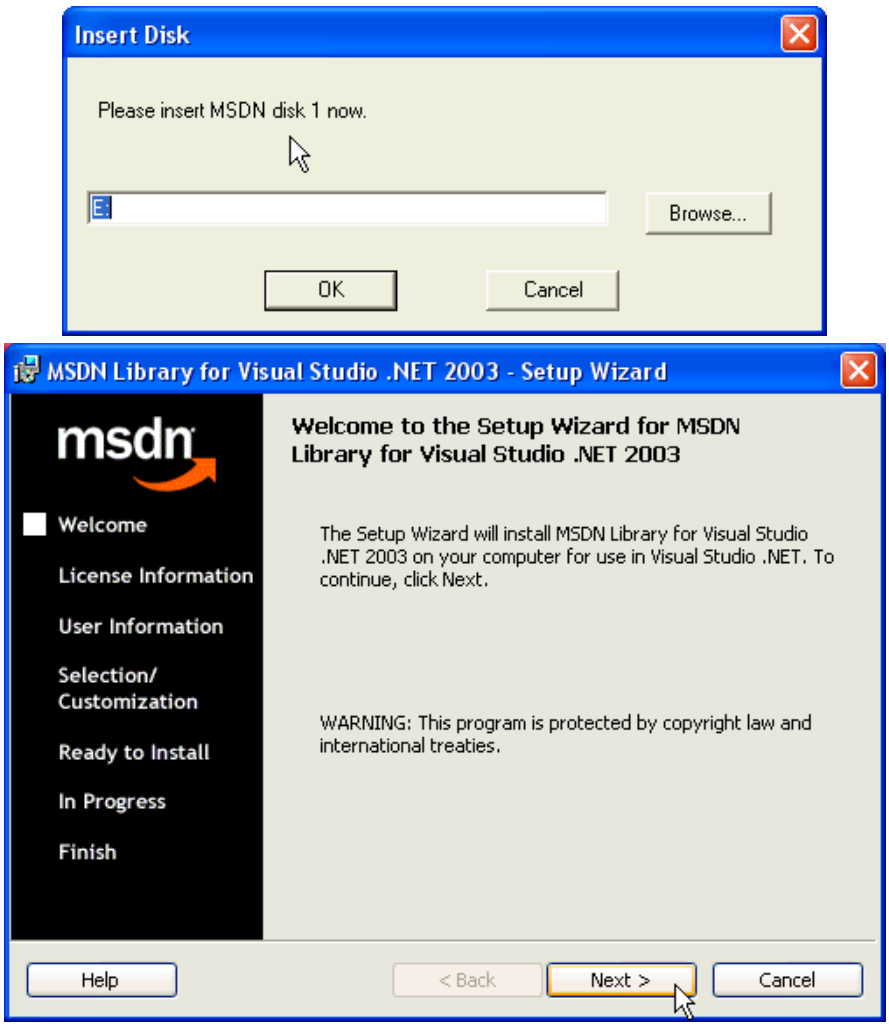


Hình 2.10: Màn hình bước 10

Bước 11:

Chọn Product Documetation và nhập dĩa 3 Microsoft Visual Studio.NET (tức dĩa 1 MSDN):

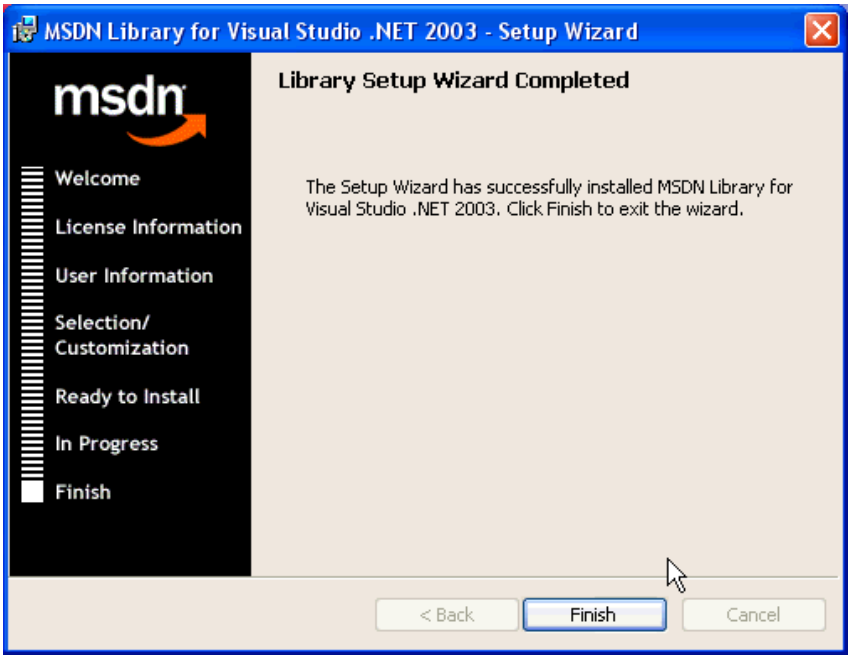




Hình 2.11: Một số màn hình bước 11

Bước 12:

Tiếp tục với các dĩa 2, 3 MSDN cho đến hết.



Hình 2.12: Màn hình bước 12

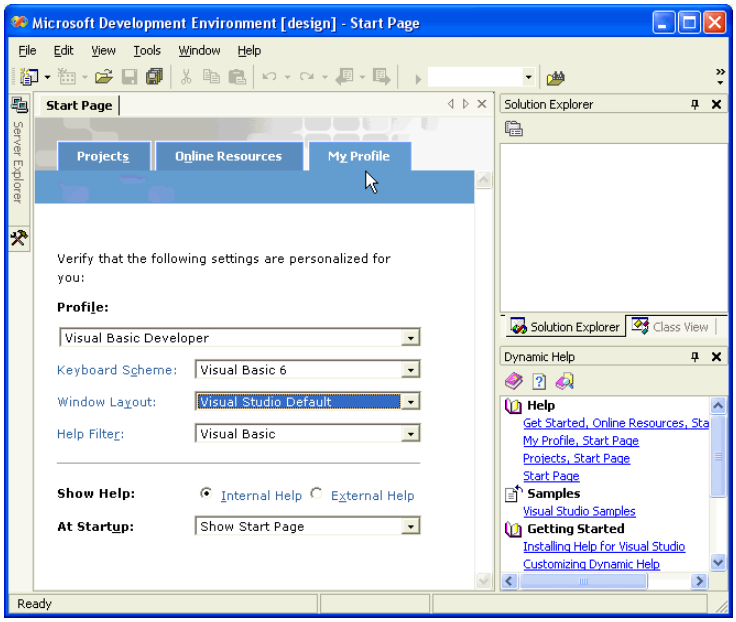
Như vậy, ta sẵn sàng cho việc lập trình với Visual Basic.NET (VB.NET). Bài kế hướng dẫn sơ lược cách dùng **Microsoft Visual Studio.NET Integrated Development Environment (gọi tắc là IDE)** cho việc tạo các ứng dụng (application) trong nền Windows.

Thật ra, ta có thể dùng Notepad để soạn mã nguồn (source code) và Visual Basic.NET compiler để chạy ứng dụng (application) mà không cần Microsoft Visual Studio.NET IDE tuy nhiên trong khóa học cơ bản, chúng tôi chọn Microsoft Visual Studio.NET để việc lập trình trở nên vui thích và hấp dẫn.

2.3 Giới thiệu MS Visual Studio.Net

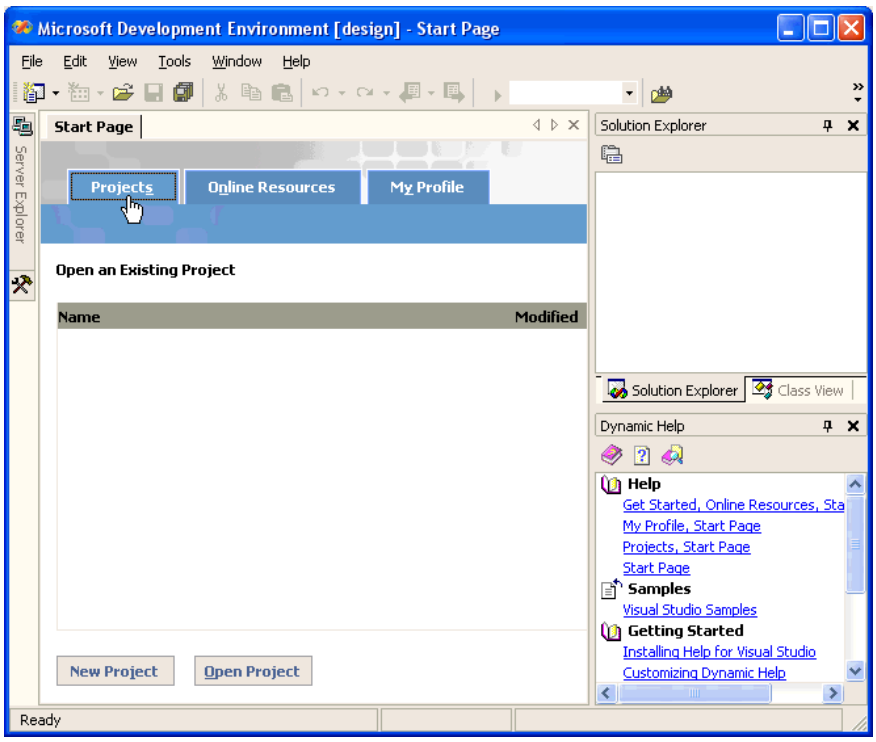
Microsoft Visual Studio.NET IDE là môi trường tập trung mọi công cụ cần thiết giúp việc lập trình dễ dàng.

- Để khởi động, chọn Start, Programs, thực đơn Microsoft Visual Studio.NET 2003 và ứng dụng (application) Microsoft Visual Studio.NET 2003.
- Chọn phần **My Profile**
- Chọn Profile là Visual Basic Developer vì khóa này chuyên trị Visual Basic.NET (VB.NET)
- Microsoft Visual Studio.NET sẽ hiển thị Visual Basic 6 trong hộp chữ **Keyboard Scheme** và ngay cả trong hộp **Windows Layout**. Bố trí này giúp tổ chức các cửa sổ trong IDE như các phiên bản trước của Microsoft Visual Studio. Trong khóa này, ta chọn Visual Studio Default.
- Bố trí gạn lọc giúp đỡ dành riêng cho ngôn ngữ lập trình Visual Basic.NET (VB.NET) trong hộp **Help Filter**.
- Internal Help hiển thị các thông tin ngay trong cùng một IDE window, trong khi External hiển thị thông tin trong 1 window riêng biệt.
- Ở phần **Startup**, chọn **Show Start Page**



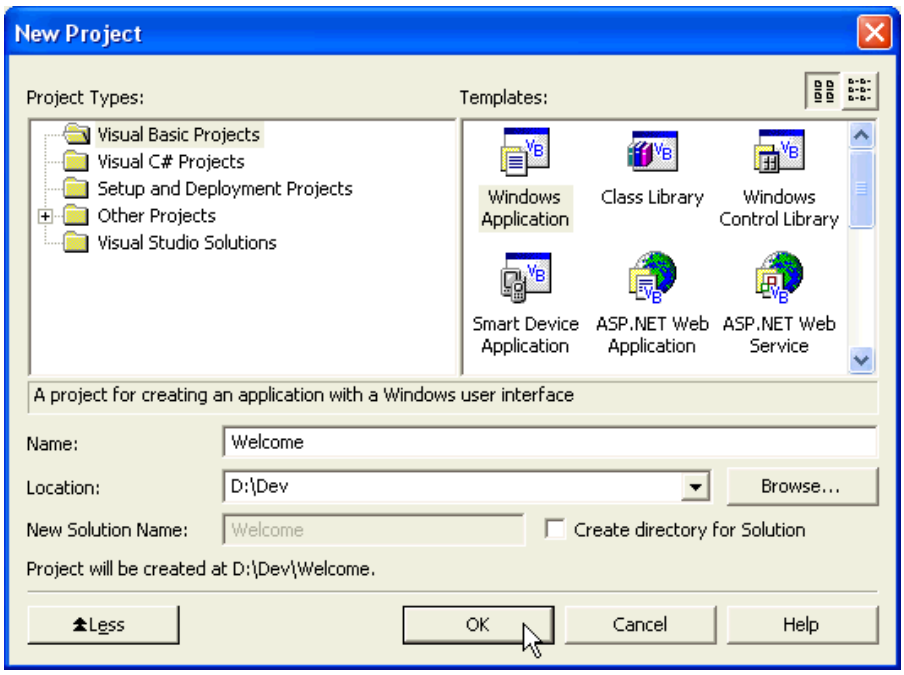
Hình 2.13: Màn hình bước 13

Đây là chổ tạo dự án mới hay mở dự án đã lập trình để sửa đổi. Ta chọn New Project để tìm hiểu thêm môi trường lập trình dùng Microsoft Visual Studio.NET



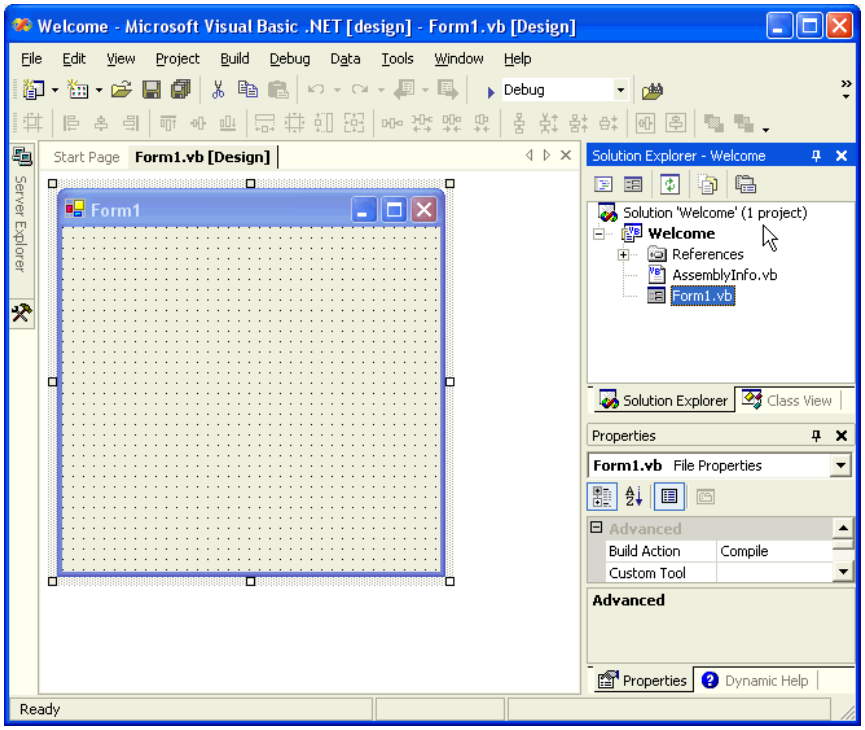
Hình 2.14: Màn hình bước 14

- Ta nhấp nút **New Project** để hiển thị bảng liệt kê các khuôn mẫu cho ứng dụng (application).
- Chọn **Visual Basic Project** trong window Project Types
- Chọn **Windows Application** trong bảng Template
- Đặt tên dự án là **Welcome**. Lưu ý ở đây, tên của dự án cũng là tên ngăn chứa (folder) chứa phụ dự trữ dự án. Thí dụ ta nhấp nút Browse để tạo 1 ngăn chứa (folder) tên Dev ở dĩa D, Microsoft Visual Studio.NET hiển thị D:\Dev ở hộp Location nhưng project sẽ được tạo và chứa ở ngăn chứa (folder) **D:\Dev\Welcome** (để ý hàng phía trên phần hiển thị các nút Less, OK, ... ta thấy hàng chữ: 'Project will be created at D:\Dev\Welcome)
- Nhấp OK



Hình 2.15: Mở dự án mới

Microsoft Visual Studio.NET IDE khởi động dự án mới trong phương thức thiết kế (Design Mode):



Hình 2.16: màn hình dự án mới

2.4 Thực đơn và thanh công cụ

Thực đơn (menu) của Microsoft Visual Studio.NET IDE ... **'biến hóa'** tùy theo công việc đang làm nhưng tổng quát, thực đơn (menu) chính hiển thị bao gồm:

File:

Tiêu chuẩn chung cho mọi ứng dụng (application) trong nền Windows. File dùng để mở (open) hay đóng (close) các tập tin (files) hay dự án (project).

Edit:

Edit cung cấp các chọn lựa khi soạn nguồn mã và dùng các công cụ lập trình, tỷ như: Undo, Redo, Cut, Copy, Paste và Delete

View:

View cung cấp sư chọn lựa hiển thị các Windows tạo môi trường của IDE, tỷ như: Solution Explorer, Properties, Output, Tool Box, Server Explorer. Nếu ta để ý sẽ thấy các Windows này thường nằm 2 bên hoặc bên dưới window thiết kế Form hay soạn nguồn mã.Các windows này cũng có thể hiển lộ hay thu kín lại nhường chổ cho window thiết kế được rộng rãi.

Project:

Dùng để quản lý dự án (project) bằng cách thêm vào hay xóa bỏ các tập tin liên hệ.

Build:

Một lựa chọn quan trọng trong thực đơn là Build cho phép ta xây dựng và chạy ứng dụng (application) 1 cách độc lập bên ngoài IDE.

Debug:

Debug không những giúp phương tiện rà tìm các lỗi lập trình trong môi trường IDE mà còn giúp kiểm tra từng bước một các nguồn mã trong dự án (project).

Data:

Giúp ta nối và sử dụng dữ kiện hay thông tin trong Cơ Sở Dữ Liệu (Database).

Tools:

Chứa các công cụ bố trí Microsoft Visual Studio.NET IDE.

Windows:

Tiêu chuẩn chung dùng quản lý mọi windows trong IDE.

Help:

Cung cấp nối yêu cầu giúp đỡ với Microsoft Visual Studio.NET documentation hay từ mạng Internet.

Cách dùng thanh công cụ sẽ được hướng dẫn tùy từng dự án (project). Tuy nhiên, 1 cách tổng quát, thanh công cụ mặc định (default) bao gồm như sau (theo thứ tự từ trái qua phải):

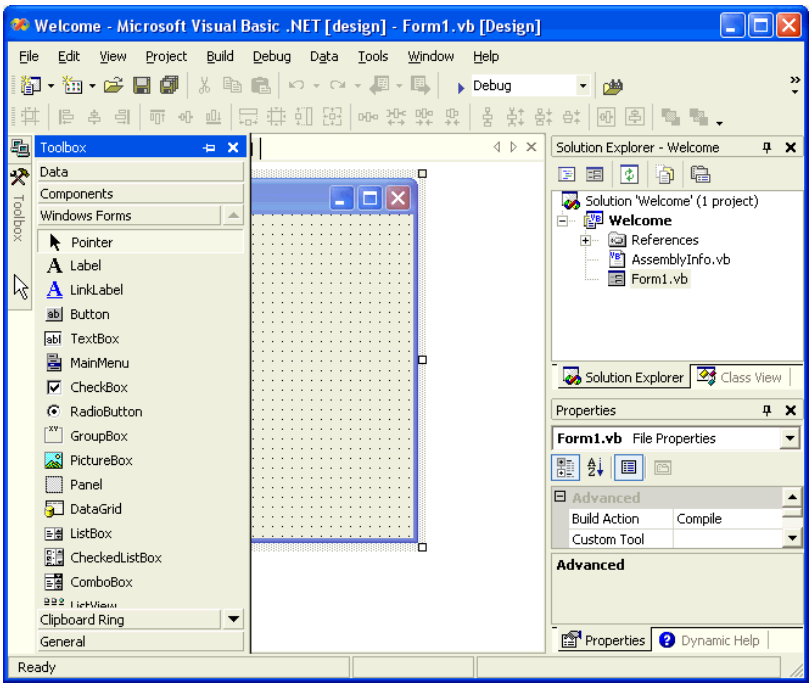
- New Project
- Add Item
- Open File
- Save (lưu trữ form hay module đang dùng)
- Save All (lưu trữ mọi forms, modules, ... đang dùng hay đang mở)
- Cut
- Copy
- Paste (sẽ hiển lộ sau khi ta nhấp nút Cut hay Copy)
- Undo
- Redo
- Navigate Backward (lướt lui)

- Navigate Forwards (lướt tới)
- Nút Start để chạy thử ứng dụng trong IDE
- Build Configuration (bố trí xây dựng ứng dụng) trong IDE. Ở đây, cho ta biết bố trí hiện dùng là Debug
- Truy tìm tập tin (Find in files)

và cuối cùng, nút Toolbar Options để hiển thị thêm các công cụ phụ thuộc khác.

Nhấp đơn hộp công cụ nằm phía bên tay trái window thiết kế như hình sau. Hộp công cụ bao gồm:

- Hộp Data
- Hộp Components
- Hộp Windows Forms
- Hộp Clipboard Ring
- Hộp Gerneral



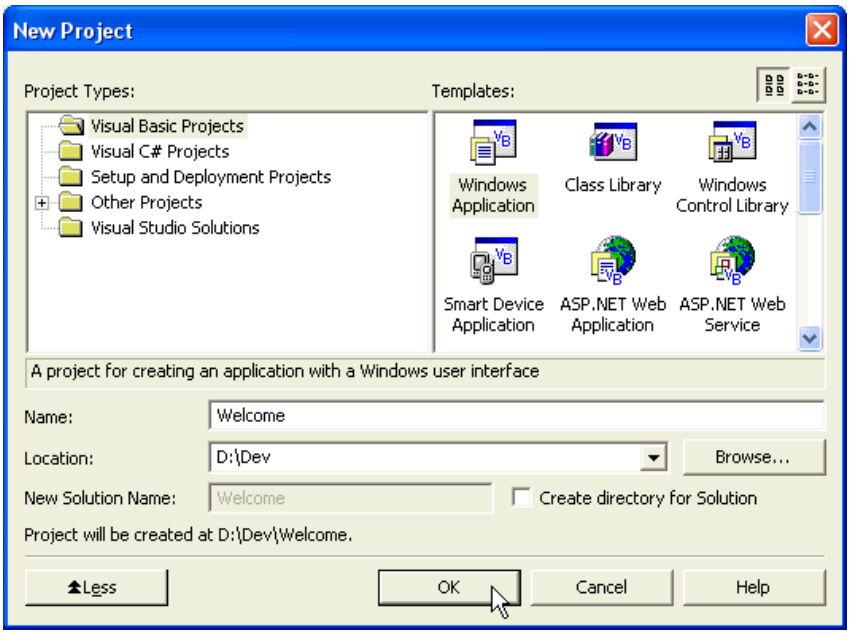
Hình 2.17: Màn hình soạn thảo

Bài kế tiếp, ta sẽ bắt đầu soạn dự án (project) đâu tiên với Microsoft Visual Studio.NET

CHƯƠNG 3: NGÔN NGỮ LẬP TRÌNH VISUAL BASIC.NET

3.1. Chương trình đầu tiên

Chúng ta bắt đầu với dự án (project) đầu tiên chào mừng các bạn đến với Visual Basic.NET. Như ta đã biết, dự án (project) Welcome được lưu trữ trong thư mục sau **D:\Dev\Welcome** như hình 2.1.



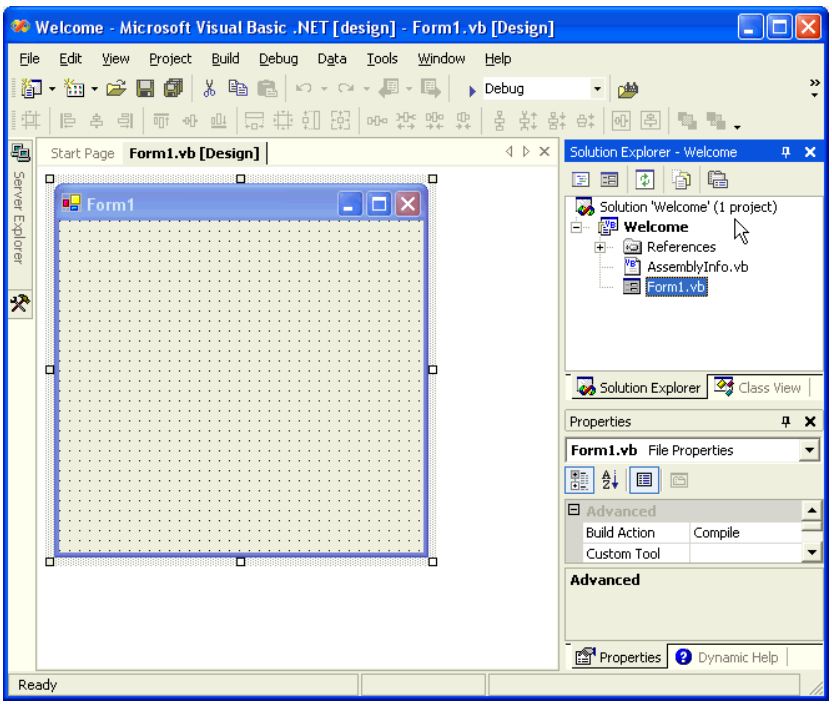
Hình 3.1. Mở dự án mới

Nhấp nút OK sẽ mở ra một cửa sổ dùng thiết kế một form trong nền Windows.

Dự án Welcome

Bước 1:

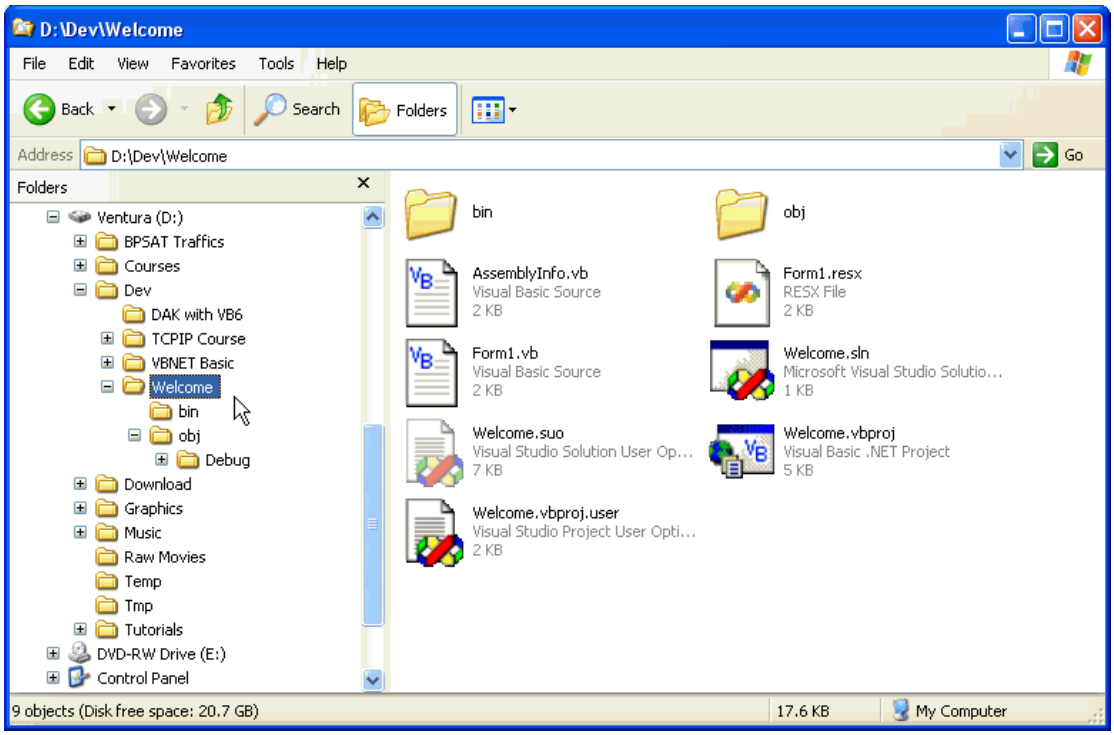
Microsoft Visual Studio.NET IDE khởi động dự án mới trong phương thức thiết kế (Design Mode) với 1 cửa sổ Form nằm ở giữa, tên mặc định là **Form1.vb**



Hình 3.2. Màn hình thiết kế

Nếu ta không lảm gì cả mà chỉ lưu trữ bằng cách chọn **File, Save All** và kiểm tra thư mục (folder) D:\Dev\Welcome, ta thấy Microsoft Visual Studio.NET tự động tạo ra và lưu trữ 1 số tập tin cần thiết trong đó có các tập tin **Welcome.sln** và **Welcome.vbproj** dùng đề quản lý dự án (project).

Thư mục (folder) **bin** là nơi lưu trữ dự án dưới hình thức ứng dụng (application) với phần đuôi là **.EXE** (ví dụ Welcome.exe) khi ta xây dựng dự án thành 1 ứng dụng (application) chạy ngoài IDE.



Hình 3.3. Các file tạo ra từ dự án

Bước 2:

- Đổi tên **Form1.vb** thành **Welcome.vb** bằng cách nhấp vào tên form ở **Solution Explorer Window** (nằm phía trên góc tay phải) hay ở hộp chữ **File Name** trong **Properties Windows** (phía dưới Solution Explorer)

Lưu ý: 1 solution có thể gồm nhiều dự án (project), 1 dự án (project) có thể gồm nhiều Forms khác nhau.

Lưu ý:

Khi đổi tên Form mặc định như vậy, ta phải bố trí **Startup Object** với tên **Welcome là object ta muốn khởi động đầu tiên** khi chạy dự án Welcome. Nếu không, dự án vẫn dùng Form1 và sẽ tạo lỗi vì Form1 đã đổi tên không còn hiện diện nữa.

- Đổi tên Form1 bằng cách chọn dự án Welcome trong **Solution Explorer** và chọn Properties.
- Chọn Welcome trong hộp chữ combo **Startup Object**.
- Nhấp nút Apply, OK

Bước 3:

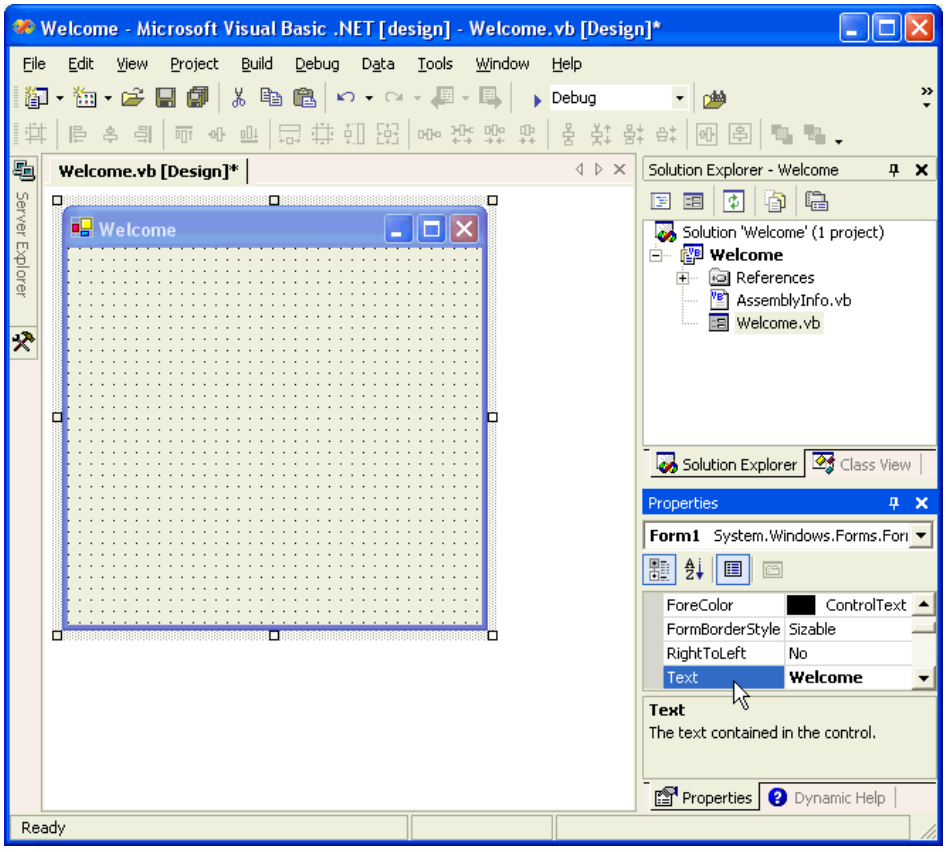
Nhấp vào Form hiển thị trong phần thiết kế.

Properties Window liên hệ thay đổi và hiển thị bảng đặc tính (properties) của Form. Bảng này sắp xếp và phân loại các đặc tính ra thành:

- **Accessibility**
- **Appearance**
- **Behaviour**

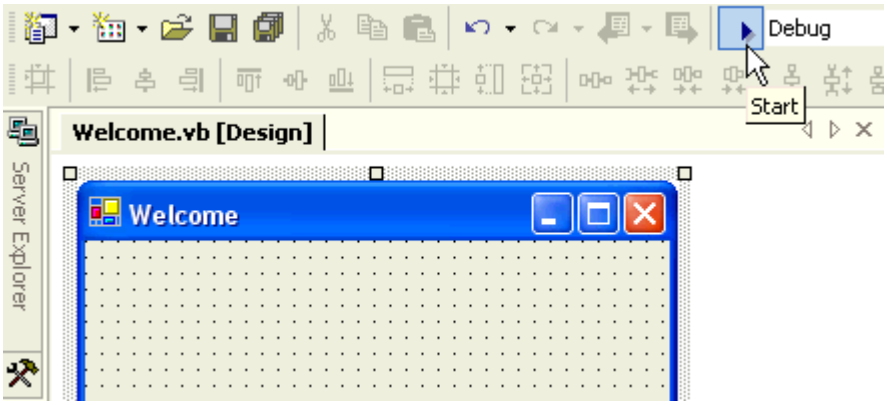
Nhằm giúp ta dễ dàng truy cập đặc tính cần đến.

Ta đổi tựa đề của Form từ Form1 ra Welcome bằng cách chọn đặc tính (property) **Text** và gỏ chữ Welcome.



Hình 3.4. Tiêu đề form

Ta có thể chọn nút Start để kiểm tra tựa đề của Form đã thay đổi theo ý hay không? Nút Start nằm ở Toolbar:



Hình 3.5. Chạy bằng nút start

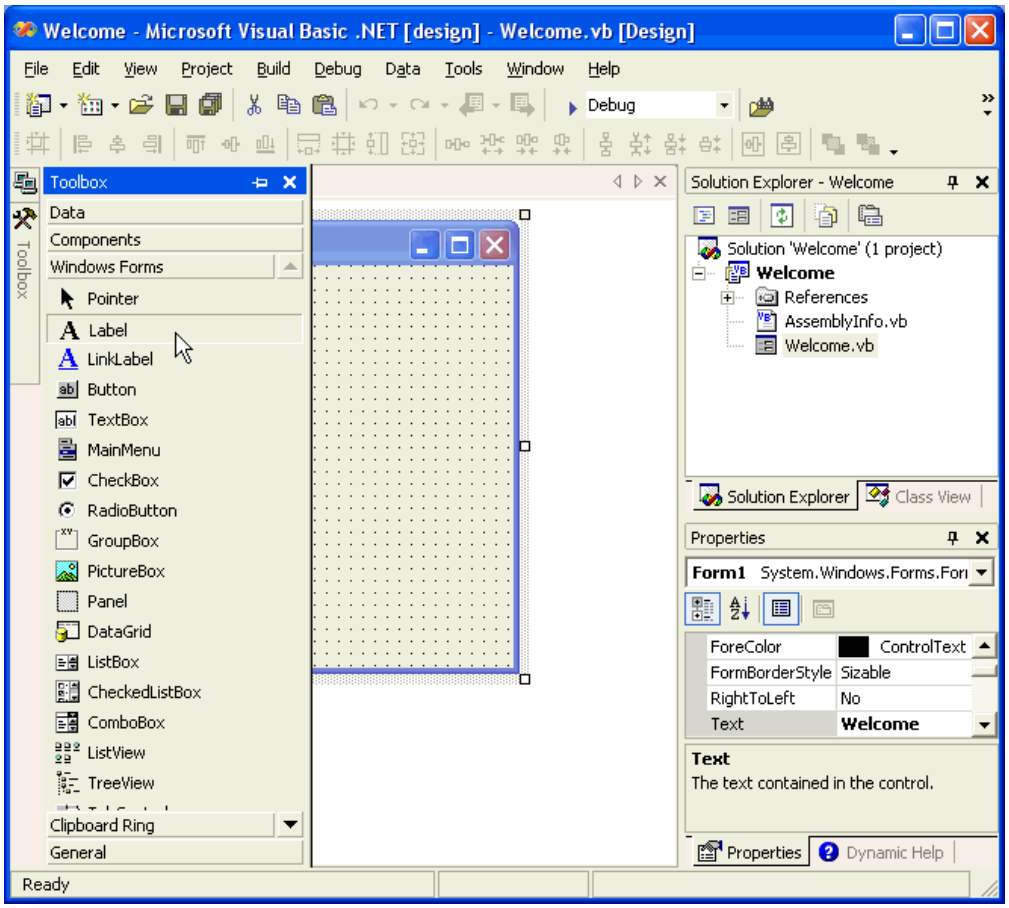
Như vậy, ta thấy Microsoft Visual Studio.NET IDE giúp ta tạo 1 Form dễ dàng như ...

Bước 4:

Từ Form 'Welcome' này, ta sẽ gắn:

- 1 nhãn hiệu (label) mang tựa 'Enter your name:"
- 1 hộp chữ để nhận dữ kiện từ user
- 1 nút mệnh lệnh 'Click Me' hiển thị hàng chữ 'Chào Mừng'
- 1 nút mệnh lệnh 'Exit' chấm dứt ứng dụng (application).

Mở Toolbox Window (nằm phía trái window thiết kế Form) và chọn công cụ **Label**. Hộp công cụ này chứa mọi đối tượng dùng tạo giao diện cũng như các công cụ phụ thuộc trong nền Windows.

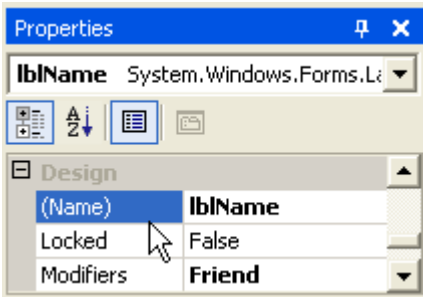


Hình 3.6. Công cụ thiết kế

Dùng mouse **kéo lê (click and drag)** 1 hình chữ nhật vừa đủ rộng nhằm chứa hàng chữ 'Enter your name:'. Nếu cần ta có thể điều chỉnh độ dài hay độ cao nhãn hiệu tùy ý.

Nhấp hộp chữ Text ở Properties Window và gỏ hàng chữ Enter your name:

Đặt tên nhãn hiệu này là **lblName** trong hộp chữ **(Name)** ở Properties Window:



Hình 3.7. Đặt tên

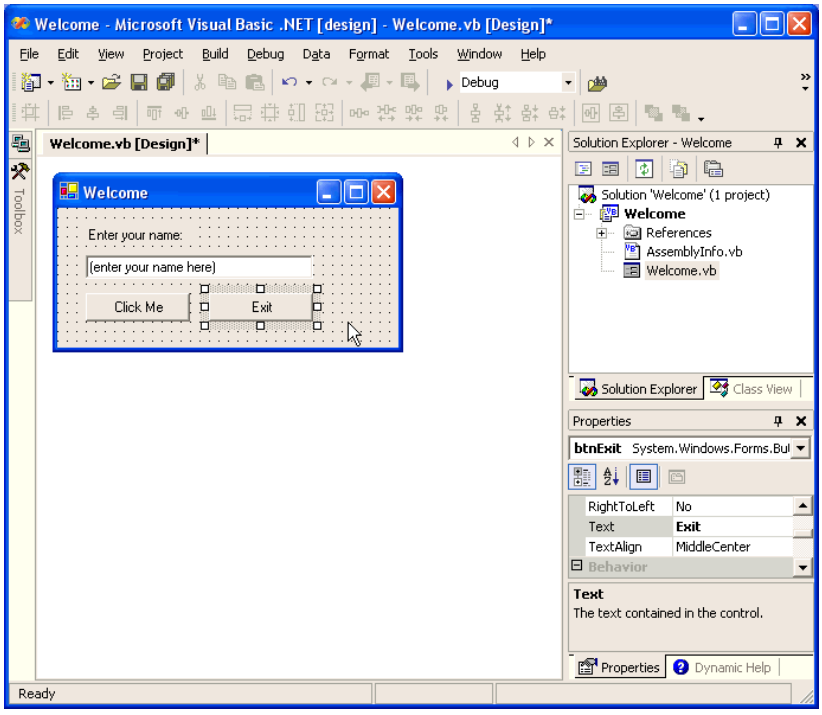
Bước 5:

Lập lại thao tác này cho các công cụ sau đây bằng cách chọn công cụ trong Toolbox và sau đó vẽ (click and drag) giao diện trên Form:

Bảng 3.1 Lựa chọn cho các đối tượng

Công cụ	Tên (Name)	Text
Textbox	tbxName	(enter your name here)
Button 1	btnClickMe	Click Me
Button 2	btnExit	Exit

Cuối cùng, ta sẽ có 1 giao diện cho ứng dụng (application) Welcome như sau:



Hình 3.8. Sau khi thiết kế

Thông thường, công ty nào cũng có tiêu chuẩn chung về danh pháp cho các hệ thống tin học, máy vi tính, thiết bị, công cụ hay nguồn mã. Để thống nhất lập trình với Visual Basic.NET (VB.NET) trong khóa học, ta có thể ấn định danh pháp cho các công cụ lập trình như sau:

Bảng 3.2: Các thuộc tính cho đối tượng form welcom

Công cụ	Tên đính kèm phía trước	Thí dụ:
Button	**btn** hoặc **cmd**	btnClickMe, cmdClickMe
ComboBox	**cbo**	cboContactName
CheckBox	**chk**	chkOver50
Label	**lbl**	lblTitle
ListBox	**lst**	lstProduct
MainMenu	**mnu**	mnuExtraOption
RadioButton	**rdb**	rdbYes
PictureBox	**pic**	picVovisoft
TextBox	**tbx**	tbxName

Như vậy, khi viết nguồn mã, mỗi lần gặp công cụ có tên đính kèm phiá trước là **tbx**, ta biết ngay đó là **Textbox**.

Bước 6:

Sau khi hoàn tất phần giao diện cho ứng dụng (application), ta cần thêm nguồn mã để xử lý các tình huống đặc biệt, tỷ như: nếu user nhấp vào nút Click Me thì chuyện gì sẽ xãy ra?

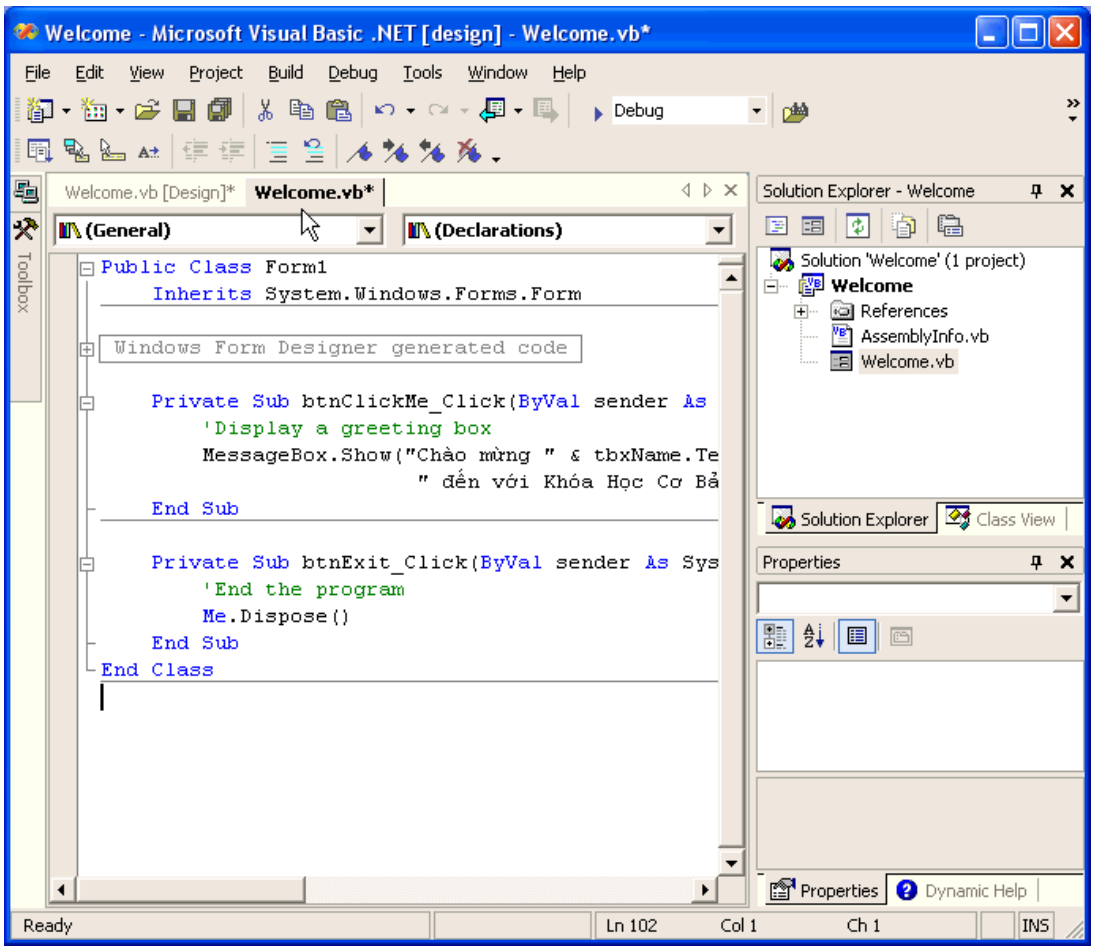
Code Editor sẽ giúp ta chuẩn bị nguồn mã. Thật vậy, khi ta nhấp đôi vào nút Click Me, Code Editor hiển thị nguồn mã tạo sẵn tổng quát cho mọi giao diện Windows và cho phép ta thêm mã vào phần **btnClickMe_Click**. Lưu ý ở đây, **Click là biến cố mặc định** khi user nhấp nút Click Me, Microsoft Visual Studio.NET chuẩn bị dùm ta 1 **Subroutine** để xử lý biến cố đó.

Lưu ý chỉ gỏ phần mã in đậm như sau:

```
Private Sub btnClickMe_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnClickMe.Click
'Display a greeting box
MessageBox.Show("Chào mừng " & tbxName.Text & _
                            " đến với Khóa Học Cơ Bản Visual
Basic.NET", "Welcome")
End Sub
```

Nhấp tab **Welcome.vb [Design]*** (kế bên tab Welcome.vb * có hình con trõ) để trở lại phần thiết kế Form, nhấp đôi nút Exit và gỏ mã:

```
'End the program
Me.Dispose()
```



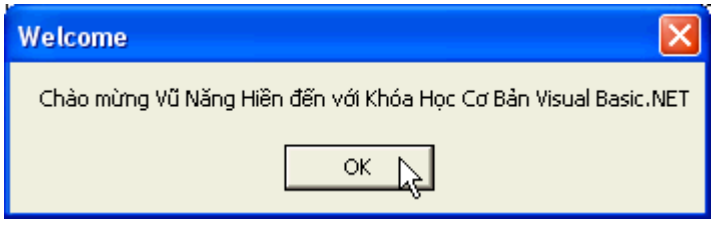
Hình 3.9. Màn hình code

Bước 7:

Nhấp nút Run để chạy thử ứng dụng trong môi trường IDE. Nhập tên vào hộp chữ dưới hàng 'Enter your name:'

Hình 3.10. Nhập liệu

Khi chọn nút Click Me, ứng dụng sẽ xử lý biến cố kích chuột đó và hiển thị 1 cửa sổ chào mừng:



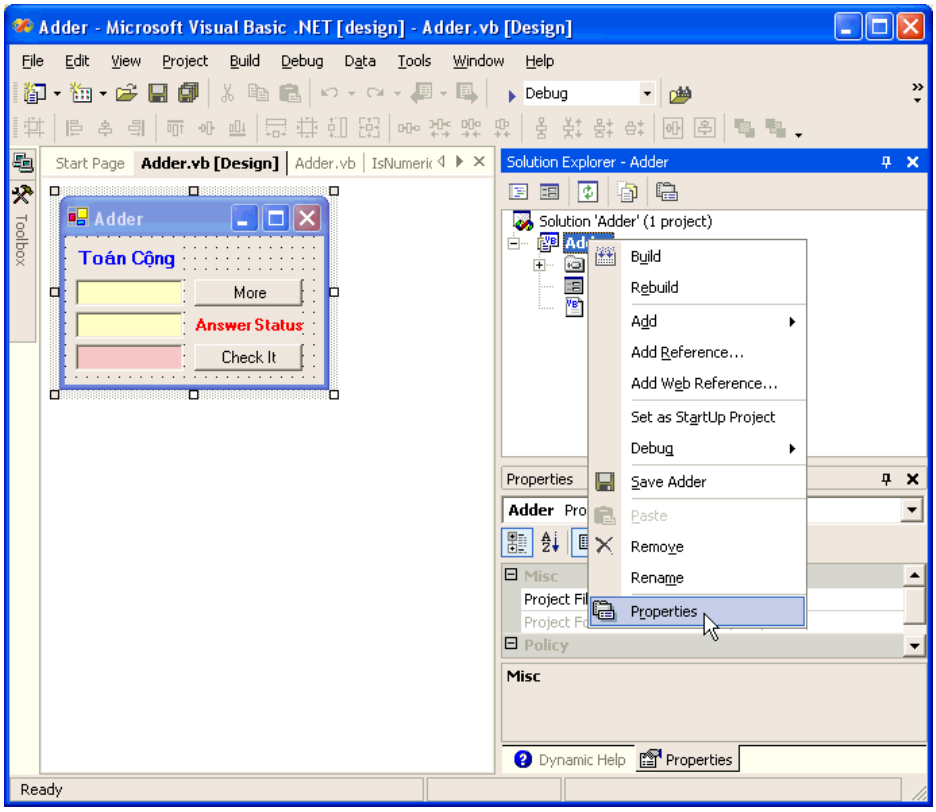
Hình 3.11. Kết quả

Như vậy, ta đã hoàn thành sứ mạng tạo 1 ứng dụng (application) đầu tiên dùng Microsoft Visual Studio.NET với ngôn ngữ lập trình Visual Basic.NET (VB.NET).

3.2. Mở rộng bài welcome

Bước 8:

Sau khi đi dạo một vòng làm quen với IDE của MS Visual Studio.NET, ta tiếp tục dự án Adder với giao diện sau:



Hình 3.12. Thiết kế chương trình

Dùng (bằng cách kéo thả - Click and Drag hay Click and Draw) các thiết bị trong hộp công cụ (Toolbox) vào Form1 và bố trí như sau:

Lưu ý, ở đây chỉ hướng dẫn và trình bày chi tiết phương pháp dùng và bố trí đặc tính (property) của 1 thiết bị trong hộp công cụ mà thôi. Sau đó, các bạn áp dụng tương tự như vậy với các thiết bị khác.

Thí dụ dùng và trình bày tiêu đề (lable) **Toán Cộng** như sau:

- Nhấp hộp công cụ (phía bên trái IDE) và nhấp đơn thiết bị **Label** (Click ...)
- Vẽ (... and Draw) 1 hình chữ nhật trong mặt trống của Form
- Chọn **Properties Window** của Label (để ý label có được chọn hay không, nếu không, ta có thể mở nhằm properties window của một thiết bị nào khác chứ không phải thiết bị ta muốn bố trí)
- Chọn **đặc tính (property) Text** va gõ hàng chữ Toán Cộng (có thể dùng ứng dụng **VPSKeys** với bố trí **Unicode** hoặc các ứng dụng gõ tiếng Việt tương đương)
- Chọn và mở rộng đặc tính (property) Fonts và thay đổi cở chữ và màu tùy ý.
- Chọn Name và đặt tên theo tiêu chuẩn định trước, tỷ như: **lblTitle** với **lbl** là chữ viết tắc của label cộng với tên của tiêu đề.
- Kéo lê (Click and Drag) thiết bị này đến vị trí tùy ý trong Form, tỷ như: vị trí phía trên bên trái như hình trình bày.

Áp dụng linh động hướng dẫn trên cho các thiết bị textbox, button, ... như sau:

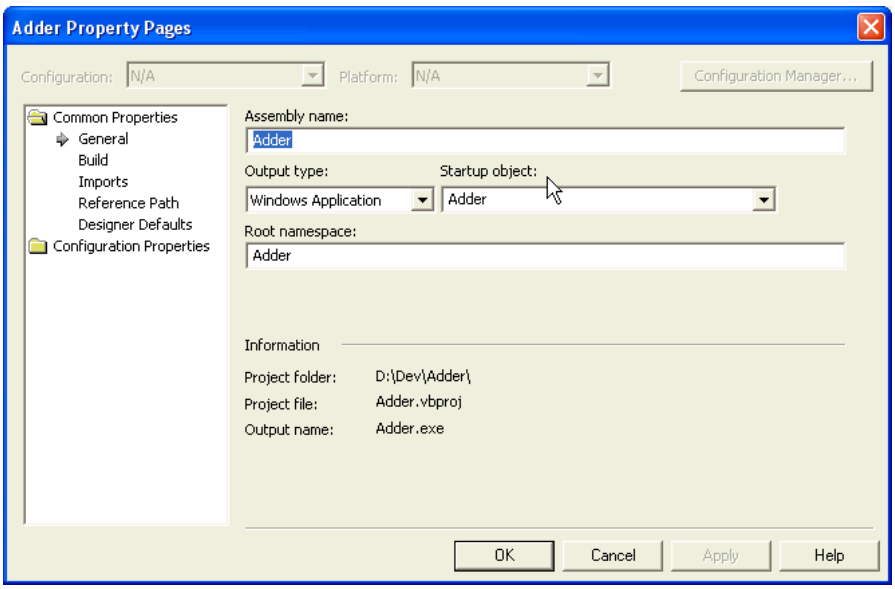
Bảng 3.3: Thuộc tính cho các đối tượng của form tính tổng

Công cụ	Bố trí đặc tính (property)
textbox1	Name = TbxNumber1 Text = (để trống ở đây) Text Align = Right BackColor = (tùy ý)
textbox2	Name = TbxNumber2 Text = (để trống ở đây) Text Align = Right BackColor = (tùy ý)
textbox3	Name = TbxNumber1 Text = (để trống ở đây) Text Align = Right BackColor = (tùy ý) ForeColor = Red
button1	Name = cmdMore Text = More
button2	Name = cmdCheckIt Text = Check It
label2	Name = lblResult Text = Answers Status TextAlign = MiddleCenter

Tuy ta có thể giữ tên mặc định Form1 trong dự án Adder nhưng có vẻ không chuyên nghiệp bằng đổi tên mặc định Form1 đó thành tên Adder thích hợp với dự án.

Lưu ý: khi đổi tên Form mặc định như vậy, ta phải bố trí **Startup Object** với tên **Adder là object ta muốn khởi động đầu tiên** khi chạy dự án Adder. Nếu không, dự án vẫn dùng Form1 và sẽ tạo lỗi vì Form1 đã đổi tên không còn hiện diện nữa.

- Đổi tên Form1 bằng cách chọn dự án Adder trong Solution Explorer và chọn Properties.
- Chọn Adder trong hộp chữ combo **Startup Object**.
- Nhấp nút Apply, OK



Hình 3.13. Thêm thuộc tính

Bước 10: Lập trình theo kiểu mẫu event - driven

Khi dùng MS Visual Studio.NET làm môi trường lập trình với Visual Basic.NET (VB.NET), thường thường ta tạo một giao diện (dưới hình thức Form) trước và sau đó gài nguồn mã vào, tỷ như: nhấp đôi nút **Check It** để mở tập tin chứa nguồn mã với tên mặc định là tên của dự án. Trước tiên, MS Visual Studio.NET sẽ tạo nguồn mã mặc định với các công dụng cơ bản yểm trợ giao diện ta vừa thiết kế (Form Adder) và ta sẽ cộng thêm mã để bố trí và kế hoạch sẵn mọi tình huống có thể xảy ra hầu hành động kịp thời tùy theo biến cố mà Form nhận được (thí dụ: người dùng nhấn vào nút Check It để kiểm tra bài toán cộng trong ứng dụng Adder). Kiểu chuẩn bị với nguồn mạ định trước như vậy được gọi là lập trình theo kiểu mẫu Event-Driven.

Bây giờ, ta bắt đầu thêm nguồn mã xử lý biến cố **Click** của nút **Check It** như sau:

- Nhấp đôi vào Form, IDE sẽ dùng **Designer Code Generator** tạo phần nguồn mã với tập tin **Adder.vb**
- Nguồn mã bắt đầu với Public Class Adder.
- Nhấp vào **tab** mang tên **Adder.vb [Design]** để trở về giao diện Form Adder. (Lưu ý hình con trỏ chỉ các tab trong IDE từ Satrt Page, Adder.vb [Design] và Adder.vb)
- Nhấp đôi vào nút **Check It** để mở phần nguồn mã của nút này với biến cố **Click**
- Gỏ nguồn mã sau đây phía dưới hàng **Private Sub cmdCheckIt_Click** (nhắc lại, **cmdCheckIt** là tên ta đặt cho nút **Check It** trong phần giao diện Form Adder): mã này kiểm tra xem ta đưa 1 giải đáp với con số hay chữ vào hộp chữ **tbxResult**? Nếu là con số, mã sẽ so sánh con số đó với kết quả bài toán cộng và báo cáo lại trong phần nhản hiệu **lblResult**.

```
Dim resultNumber As Integer
    If IsNumeric(tbxResult.Text) Then
    resultNumber = CInt(tbxNumber1.Text) + CInt(tbxNumber2.Text)
    If CInt(tbxResult.Text) = resultNumber Then
      lblResult.Text = "Correct"
    Else
      lblResult.Text = "Wrong"
    End If
```

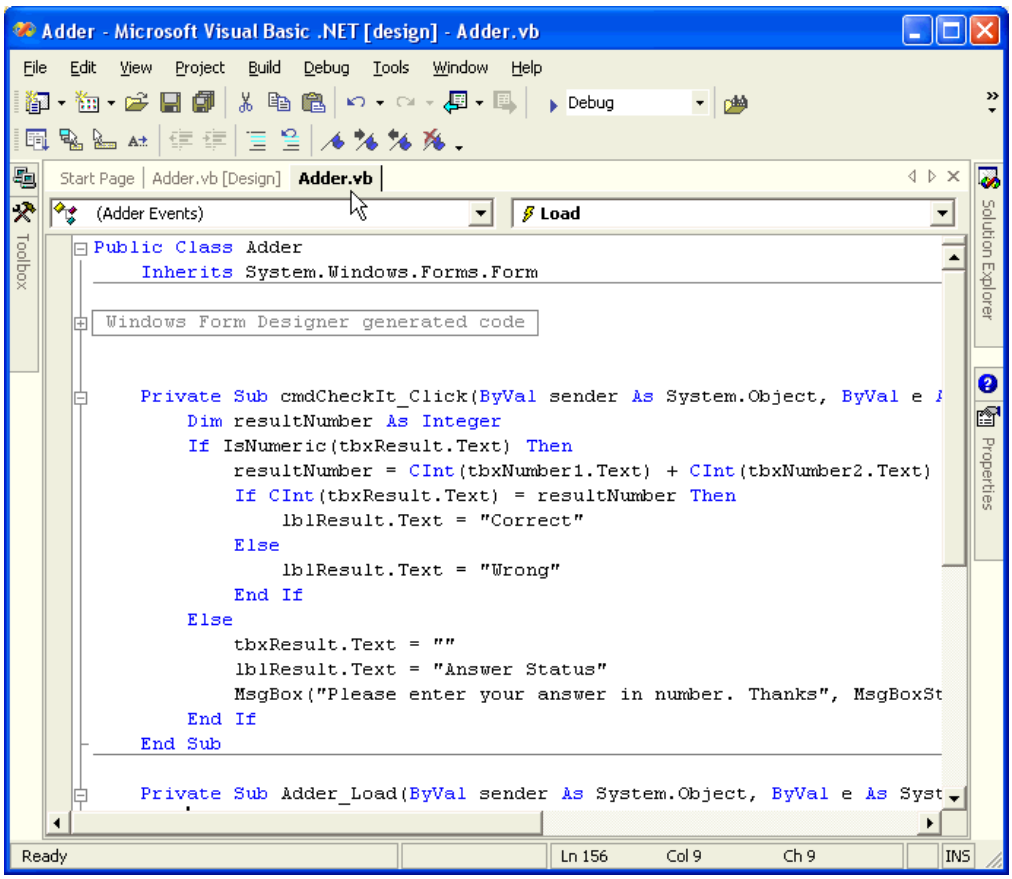
```
    Else
      tbxResult.Text = ""
      lblResult.Text = "Answer Status"
      MsgBox("Please enter your answer in number. Thanks",
MsgBoxStyle.Information, "Warning")
      End If
```

Tương tự, trở về phần thiết kế Form:

- Nhấp đôi vào chổ trống của Form cho nguồn mã Adder_Load
- Nhấp đôi vào nút More cho nguồn mã cmdMore_Click
- Gỏ nguồn mã cho Subroutine (sẽ học cách tạo Subroutine và Function ở các bài kế) SetRandomNumber. Mã ở đây tạo 2 con số ngẫu nhiên từ 1 đến 10000 cho bài toán cộng khi chạy ứng dụng Adder trong phần **Adder_Load** và trong nút **More**.

```
        Private Sub Adder_Load(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles MyBase.Load
            SetRandomNumber()
        End Sub
        Private Sub cmdMore_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles cmdMore.Click
            SetRandomNumber()
        End Sub
        Private Sub SetRandomNumber()
            Dim firstNumber, secondNumber As Integer
            Randomize()
            firstNumber = CInt(Int((10000 - 0 + 1) * Rnd() + 0))
            secondNumber = CInt(Int((10000 - 0 + 1) * Rnd() + 0))
            tbxNumber1.Text = firstNumber
            tbxNumber2.Text = secondNumber
        End Sub
```

Hình đặc trưng nguồn mã của dự án Adder:



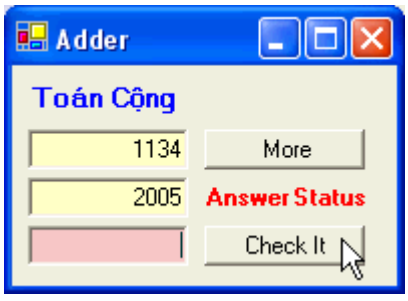
Hình 3.14. Viết code

Bước 11:

Nhấp nút Run (như hình dưới đây) để chạy ứng dụng (application) Adder trong môi trường IDE:

Hình 3.15. Thanh menu

Ta thấy bài toán cộng được hình thành với 2 số ngẫu nhiên và chờ ta gỏ vào giải đáp trong hộp chữ kế bên bút Check It. Sau đó, ta nhấp nút này để kiểm tra kết quả. Khi nào muốn làm lại bài toán cộng này, nhấp nút More:



Hình 3.16. Chạy chương trình

Lưu ý: MS Visual Studio.NET tạo một executable file mặc định là Adder.exe dưới một ngăn chứa cấp dưới (subfolder) **BIN.** Tập tin này là ứng dụng Adder tạo ra bởi dự án Adder.

Bước 12:

Lưu trữ mọi tập tin với thực đơn **File, Save All**

3.3. Dữ liệu và biến

Thông tin (Information) diễn tả một sự việc nào đó dưới nhiều hình thức khác nhau, tỷ như: tin tức trên báo, tin nhận được từ ký giả viết tay trên giấy, sự cố báo cáo trên TV, ... khác với dữ kiện (Data) dùng diễn tả thông tin đã được kiểm tra, đối chiếu, so sánh, xếp loại theo thứ tự và quan trọng hơn cả là được tổ chức để dùng trong một ứng dụng (application) điện toán. Do đó, thông tin được ghi chú ở các sổ tay không thể là dữ kiện mà một ứng dụng (application) nào đó có thể dùng được. Nếu muốn dùng thông tin như vậy, ta phải chuyển đổi qua hình thức dữ kiện, tỷ như: rà (scan) hay nhập (enter) vào 1 trang kế toán của MS Excel để có thể phân tích kết quả thu lượm.

Mặc dù, Công Nghệ Tin Học đã phát triển và thay đổi nhanh chóng nhưng tiến trình xử lý và phát triển nhu liệu hầu như vẫn ... 'trước sau như một', nghĩa là không đổi gì cả. Ở đây, ta muốn nói đến **phương thức cơ bản** cho phát triển và giải đáp vấn đề cho việc lập trình. Anh Ngữ gọi là **Algorithm**. Algorithm đó là:

Trước khi ta viết nhu liệu giải quyết một vần đề nào đó, ta phải phân ra (phân tích) thành những phần nhỏ hơn tùy từng trường hợp một để diễn tả cách giải quyết vấn đề và sau cùng tổng hợp lại. Tóm lại, đây là một **phương thức phân tích tổng hợp**. Nếu không áp dụng phương thức này, vấn đề xem có vẻ như ... 'rối tung lên' không thể giải quyết được.

Bây giờ, tưởng tượng bạn đang làm việc cho một ông ty viễn thông. Vấn đề đặt ra là làm sao cung cấp được hoá đơn tính tiền điện thoại mà khách hành đã dùng. Ta phải bắt đầu từ đâu? Làm gì trước, làm gì sau? Hoá đơn như thế nào? ...

Phương thức cần có là chia vấn đề thành những phần việc nhỏ và truy cập cách giải quyết phần việc đó, giả dụ như:

- Vào mỗi đầu tháng, ta sẽ cung cấp hoá đơn đến mỗi khách hàng.
- Cho mỗi khách hành, ta cần một bảng liệt kê các cú gọi đi trong tháng.
- Ta cần biết khoãng thời gian dùng cho mỗi cú điện thoại? lúc gọi? trong tuần hay cuối tuần? ban ngày hay ban đêm? để tính toán chi phí mỗi cú điện thoại.
- Trong từng hoá đơn một, ta tổng kết chi phí các cú điện thoại (dưới tiêu đề nội địa, ngoại quốc hay mobile, ...).
- Trong các dịp lễ lạc hay khuyến mãi, bao nhiêu phần trăm hạ giá?
- Ta cần cộng thêm tiền thuế bán dịch vụ cho mỗi hoá đơn.
- Sau khi tổng hợp lại, in ra và gởi hoá đơn đến khách hàng.

Như vậy, ta thấy phân tích để giải quyết vấn đề khi viết nhu liệu, ta hoàn toàn không để ý hay làm gì dính dáng tới ngôn ngữ lập trình. Thật sự, đây là mấu chốt quan trọng nhất của một chuyên gia lập trình chuyên nghiệp. Nếu không, ta chỉ là ... thiên lôi, ai sai đâu thì ... đánh đó, không thể tự mình đưa giải đáp cho các trở ngại nêu ra trong khi chuẩn bị thiết kế và phát triển một ứng dụng (application). Nên làm chuyên gia lập trình chứ đừng ngừng lại ở ... 'người viết mã' mà thôi.

Việc còn lại là chọn cho mình một ngôn ngữ lập trình hùng mạnh đủ khả năng phát triển các giải đáp cho mọi trở ngại: Visual Basic.NET (VB.NET).

Một cách tổng quát, ngôn ngữ lập trình chỉ gồm các **biến số (variables) và cách thức (methods)**. Ngôn ngữ lập trình dù phức tạp đến đâu thì cũng được xây dựng trên các biến số và cách thức mà thôi. Do đó, ta không thể so sánh ngôn ngữ lập trình này mạnh hơn hay yếu hơn, nhất là các ngôn ngữ lập trình .NET như Visual Basic.NET (VB.NET) hay C# hay C++.

Trên thực tế, các ngôn ngữ lập trình .NET đều được biên dịch ra một ngôn ngữ trung gian là MSIL (Microsoft Intermediate Language).

Ngôn ngữ lập trình chỉ là công cụ phụ giúp công việc của ta và chắc chắn sẽ thay đổi trong tương lai.

3.4. Biến số (Variable)

Biến số (Variable) dùng chứa một giá trị nào đó trong phương thức lập trình (algorithm). Ta có thể làm một quyết định dựa trên giá trị đó, ví dụ: giá trị đó bằng 9 không? hay nhỏ hơn 7? hay có thể thực hiện các thuật toán trên giá trị đó như cộng, trừ, nhân, chia, ...

Xét phương thức lập trình (algorithm) sau:

- Tạo 1 biến số đặt tên là 'count'
- Trong biến số count, chứa giá trị 35
- Cộng thêm 1 vào biến số count
- Hiển thị giá trị của biến số count trên màn hình (monitor)

Như vậy, ta phải tuyên bố biến số (variables) count, cho vào giá trị 35, cộng 1 thành 36 và hiển thị số 36 trên màn hình.

Trong Visual Basic.NET (VB.NET), dùng **Dim** và **Redim** tuyên bố biến số như sau:

```
Dim myVariable As Long
Dim myArray (5) As Integer
Dim yourArray ( ) As String = {"Dần", "Thân", "Tỵ", "Hợi",
"Tứ Hành Xung"}
Redim myArray (10) As Integer
```

Giải thích:

Dùng **Dim** tuyên bố (hay tuyên cáo) biến số myVariable thuộc loại dữ kiện **Long**. Redim để tuyên bố lại, nhất là khi thay đổi cở của **Array**. myArray (5) là một chuỗi biến số gồm 6 số bắt đầu từ số 0 với myARray (0), myArray (1), ... đến myArray (5) loại dữ kiện số nguyên (Integer). yourArray ( ) dùng giá trị bên trong dấu { } để xác định cở (ở đây, cở = 5, chỉ số hay 'index' **bắt đầu từ số 0**, 1, 2, 3, 4).

Array:

Array dùng chỉ số (index) để lưu trữ nhiều giá trị dưới cùng một tên biến số (variables), tỷ như:

```
Dim yourArray ( ) As String = {"Dần", "Thân", "Tỵ", "Hợi",
"Tứ Hành Xung"}
Dim strMonths ( ) As String = {"Giêng", "Hai", "Ba", "Tư",
"Năm", "Sáu", "Bảy", "Tám", "Chín", "Mười", "Mười Một",
"Chạp"}
Dim empRecords (100)
```

3.4.1. Chú thích

Trình biên dịch Visual Basic.NET (VB.NET Compiler) bỏ qua không biên dịch các phần **comments**, do đó ta có thể chú thích thêm phần dẫn giải hay phương thức giải quyết vấn đề cho từng nguồn mã. Chuyên nghiệp nhất là ghi lại algorithm của ta để các lập trình viên khác hay ... cả chính ta có thể hiểu mã ta đã viết .. từ nhiều tháng trước. Nhớ là con người

cũng ... 'mau quên lạ lùng'. Trên thực tế, chính ta cũng không biết ta viết .... cái gì nếu đọc lại mã sau ... chừng vài tháng.

Trong Visual Basic.NET (VB.NET), đánh dấu nơi ghi chú thích với **dấu ' (dấu apostrophe)** , tỷ như:

```
        'tạo biến số count và chứa giá trị 35
    Dim count As Integer
    count = 35
         'cộng thêm 1 vào count
    count = count + 1
         'hiển thị giá trị của count
    MessageBox.Show ("Value of count is now " & count)
```

Whitespace cũng quan trọng không kém. Việc chừa các khoãng trống như vậy nhằm cho nguồn mã được đọc dễ dàng. Thường thường, ta nên chừa một hàng trống giữa các bước trong phươn gthức lập trình (algorithm) như thí dụ trình bày ở trên, ta thấy có hàng trống sau hàng count = 35.

3.4.2. Loại dữ liệu (Data Types)

Khi dùng biến số (variables), ta cần biết và bố trí trước biến số đó lưu trữ loại dữ kiện (data types) nào, điều này giúp ích máy vi tính xử lý tài nguyên dễ dàng hơn trong lúc chạy ứng dụng (application).

Tổng quát, các loại dữ kiện (data types) bao gồm:

Bảng 3.4: Số nguyên (Number)

Loại dữ kiện	**Cở (Size)**	**Range**	**Chú thích**
Byte	1 byte	0 tới 255	Byte = 8 bits trong hệ thống nhị phân. Byte không yểm trợ số âm (negative number).
Short	2 bytes	-32,768 tới 32,768	Rất tiện lợi cho các biến số (variables) lưu trữ số nguyên cở nhỏ.
Integer	4 bytes	-2,147,483,648 tới 2,147,483,647	Tiêu chuẩn số nguyên. Loại dữ kiện này được máy vi tính xử lý nhanh nhất và ít tài nguyên nhất.
Long	8 bytes	9,223,372,036,854,775,808 tới 9,223,372,036,854,775,808	Đây là số nguyên lớn từ -9 quintillion tới 9 quintillion (-9 x 10^{18} tới +9 x 10^{18})

Bảng 3.5: Số thực (Decimal Number)

Loại dữ kiện	**Cở (Size)**	**Range**	**Chú thích**
Single	4 bytes	Cho số âm: -3.402823 x 10^{-38} tới -1.401298 x 10^{-45}. Cho số dương: 1.401298 x 10^{-45} tới 3.402823 x 10^{38}.	Đây là số thực vô cùng nhỏ hay vô cùng lớn.

Double	8 bytes	Cho số âm: -1.79769313486231 x 10^{308} tới -4.94065645841247 x 10^{-324}. Cho số dương: 4.94065645841247 x 10^{-324} tới 1.79769313486231 x 10^{308}.	Double còn gọi là loại dữ kiện 'double prescision floating point' do có thể lưu trữ số lẻ gấp đôi loại 'single', tức là 15 số lẻ sau '**decimal point**'.

Bảng 3.6: Chữ và hàng chữ (hay câu)

Loại dữ kiện	Cở (Size)	Range	Chú thích
Char	2 bytes	Một chữ	Dùng lưu trữ từng chữ một.
String	10 bytes + 2 bytes cho mỗi chữ (character)	Hàng chữ có thể kéo dài tới 2 tỷ (billion) chữ	Dùng lưu trữ một hàng chữ hay cả nguyên một cuốn sách.

Bảng 3.7: Các loại đơn giản khác

Loại dữ kiện	Cở (Size)	Range	Chú thích
Boolean	2 bytes	True hoặc False	VB.NET dùng 2 bytes cho số 0 (False) và 1 (True).
Date	8 bytes	Từ ngày 1 tháng Giêng năm 100 tới ngày 31 tháng Chạp năm 9999	Loại dữ kiện có khả năng tính toán năm nhuần. Nếu ta cộng 1 ngày vào biến số lưu trữ ngày 28/02/2000, ta sẽ có 29/02/2000 nhưng nếu cộng cho ngày 28/02/2001, ta lại có 01/03/2001.

3.4.3. Hằng số (Contants)

Trái với biến số (variables), hằng số không thay đổi giá trị trong suốt đời sống của ứng dụng (application). Ta dùng **Const** để tuyên bố hằng số, tỷ như:

```
Const PI = 3.1416 As Double
Const DSN As String = "MyDatabaseName"
```

3.5. Tên

Thông thường, ta thoả thuận một danh pháp chung khi đặt tên các biến số (variables) hay hằng số, nếu không, chính ta sau này có thể mất công tìm hiểu loại các biến số hay hằng số trong ứng dụng (application). Quy ước tổng quát khi đặt tên bao gồm 2 phần:

- Tiền tố (Prefix): thường dùng chữ in thường chỉ loại biến số (variables) hay hằng số (constant).
- Tên: chữ đầu tiên dùng chữ Hoa và tên phải đầy đủ ý nghĩa để khỏi mất công tham khảo sau này.

Đề nghị tên quy ước như sau:

Bảng 3.8: Qui ước đặt tên

Loại dữ kiện	Tiền Tố (Prefix)	Thí dụ
Byte	Byt	bytAge
Short	Sht	shtCounter

Integer	Int	intCounter
Long	Lng	lngResolution
Single	Sng	sngInterestRate
Double	Dbl	dblTotalSalesInYear
Char	Chr	chrMiddleName
String	Str	strAddress
Boolean	Bol	bolIsCompleted
Date	Dte	dteHireDate
(User-defined types)	(vài chữ - 2 hay 3 chữ trong tên của structure)	empRecord
Constant	(no prefix, chữ nối nhau bằng dấu _ **underscores**)	TAX_RATE
Enumerations	(vài chữ - 2 hay 3 chữ trong tên của Enumerations)	dteWeekday, dowWeekday, colBackGroundColor

3.6. Phương thức (method)

Method là nguồn mã độc lập (self-contained) dùng để thực hiện công việc ta muốn làm trong ứng dụng (application). Method rất quan trọng vì:

- Phân giải (break-up) chương trình thành các phần tử nhỏ hơn, có trách nhiệm rõ ràng, đơn giản hơn và dễ hiểu.
- Khuyến khích dùng lại nguồn mã (reusable code)

Ta phân biệt 2 loại mehods:

- `Subroutine: với Sub ... End Sub`
- `Function: với Function ... End Function`

Dùng Subroutine khi ta muốn thực hiện công việc gì đó và Function khi muốn nhận kết quả trả về.

Thí dụ dùng subroutine 'SetRandomNumber' trong bài 5:

Bố trí 2 số ngẫu nhiên vào hộp chữ nhưng không trả về giá trị nào.

```
Sub SetRandomNumber()
  Dim firstNumber, secondNumber As Integer
  Randomize()
  firstNumber = CInt(Int((10000 - 0 + 1) * Rnd() + 0))
  secondNumber = CInt(Int((10000 - 0 + 1) * Rnd() + 0))
  tbxNumber1.Text = firstNumber
  tbxNumber2.Text = secondNumber
End Sub
```

Thí dụ dùng Function:

Cộng 2 số và trả về kết quả bài toán cộng.

```
Function Addition (ByVal Number1 As Integer, ByVal Number2 As Integer)
    Addition = Number1 + Number2
End Function
```

Tên methods:

Thường thường, ta đặt tên quy ước cho methods bằng động từ chỉ công việc thực hiện và tên phải đầy đủ ý nghĩa, Ví dụ:

```
GetCustomerName
OpenCustomerRecord
CalculateRepaymentPerMonth
ReadXMLFile
GetEnvironementVariables
SaveMyNetworkConfiguration
```

3.7. Phạm vi (scope)

Phương thức (method) là nguồn mã chạy độc lập, do đó các biến số (variables) được khai báo trong một method chỉ có ý nghĩa khi dùng trong method đó mà thôi. Ta gọi là trong phạm vi method (**scope)**. Biến số (variables) dùng trong method này không có ảnh hưởng gì đến biến số (variables) trong method khác, ví dụ:

```
  Sub DisplayMyName
        Dim strName
        strName = "Vũ Năng Hiền"
        MessageBox.Show(strName)
    End Sub
    Sub DisplayYourName
        Dim strName
        strName = "Đặng Quang Lương"
        MessageBox.Show(strName)
    End Sub
```

Ta nhận thấy 2 subroutine có cùng 1 biến số (variables) **strName** nhưng giá trị 2 biến số (variables) này khác nhau. Thay đổi giá trị biến số **strName** trong subroutine **DisplayMyName** không làm thay đổi giá trị biến số strName trong subroutine **DisplayYourName**.