Before the start

- Instructor: Fan Wu, kfwukfwu@gmail.com
- Class time ：
  - at R004/101, College of management (for lecture)
  - at R349, College of management (for hands-on)
- Office hours: Mon.-Wed., by appointment
- **Course web page:** elearning.ecourse.ccu.edu.tw
  - Up to date information
  - Lecture notes
  - Relevant dates, links, etc.
- TAs: 林家賢
  - Email: f5822538@gmail.com
  - Cell phone: 0952125032

◀ ▶

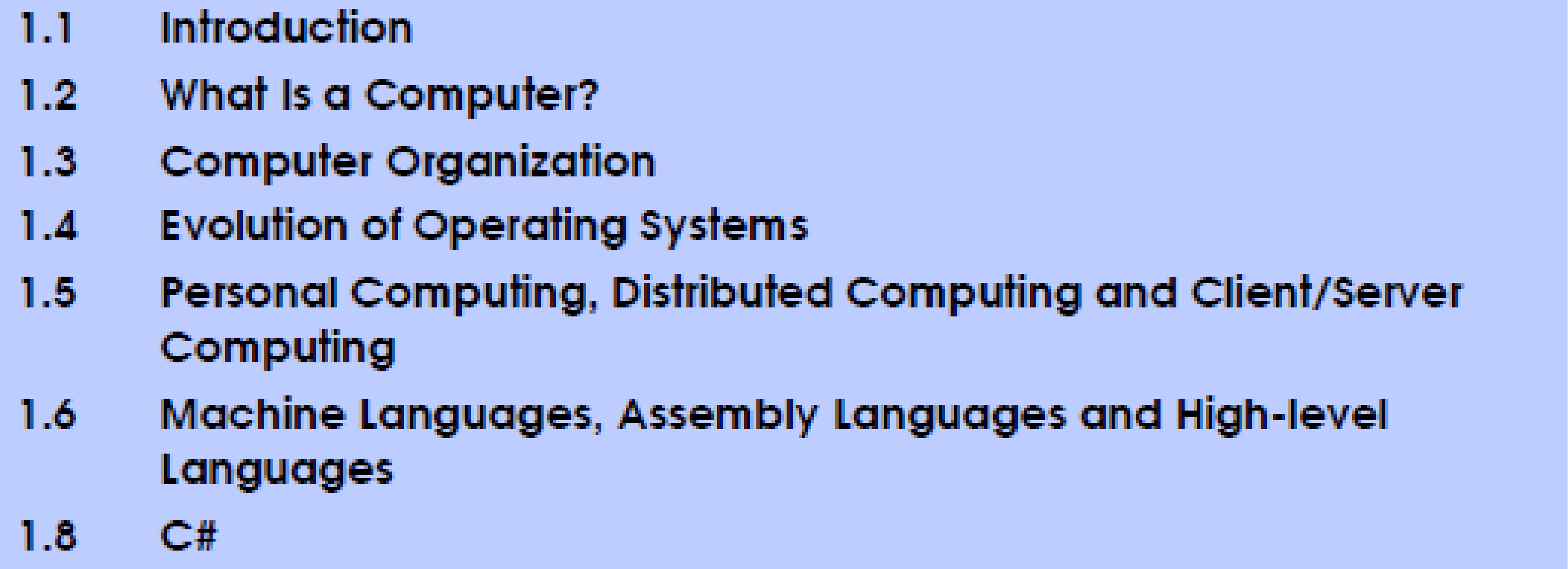
Before the start

- Course material:
  - Textbook: **C#, How to Program, by Deitel**
    - **It can be found free version through internet**
  - Hand out
- Prerequisites:
  - Working hard, clear brain, good memory

Before the start

- **Grading (not 100% for SURE):**
  - 30% for midterm exam + 30% for final exam + 25% exercise + 15% for Quiz (the attendants for a quiz will get at least 70 points for the quiz)
  - 10% bonus for acceptable interaction in class
- Question: How can I pass in the course?
  - Writing the program by yourself;
  - Ask senior schoolmates to help you to fully understand the program
  - Don’t be late for the class
  - Read the textbook of English version

Chapter 1 – Introduction to Computers, the Internet, the Web and C#



1.1 Introduction

1.2 What Is a Computer?

1.3 Computer Organization

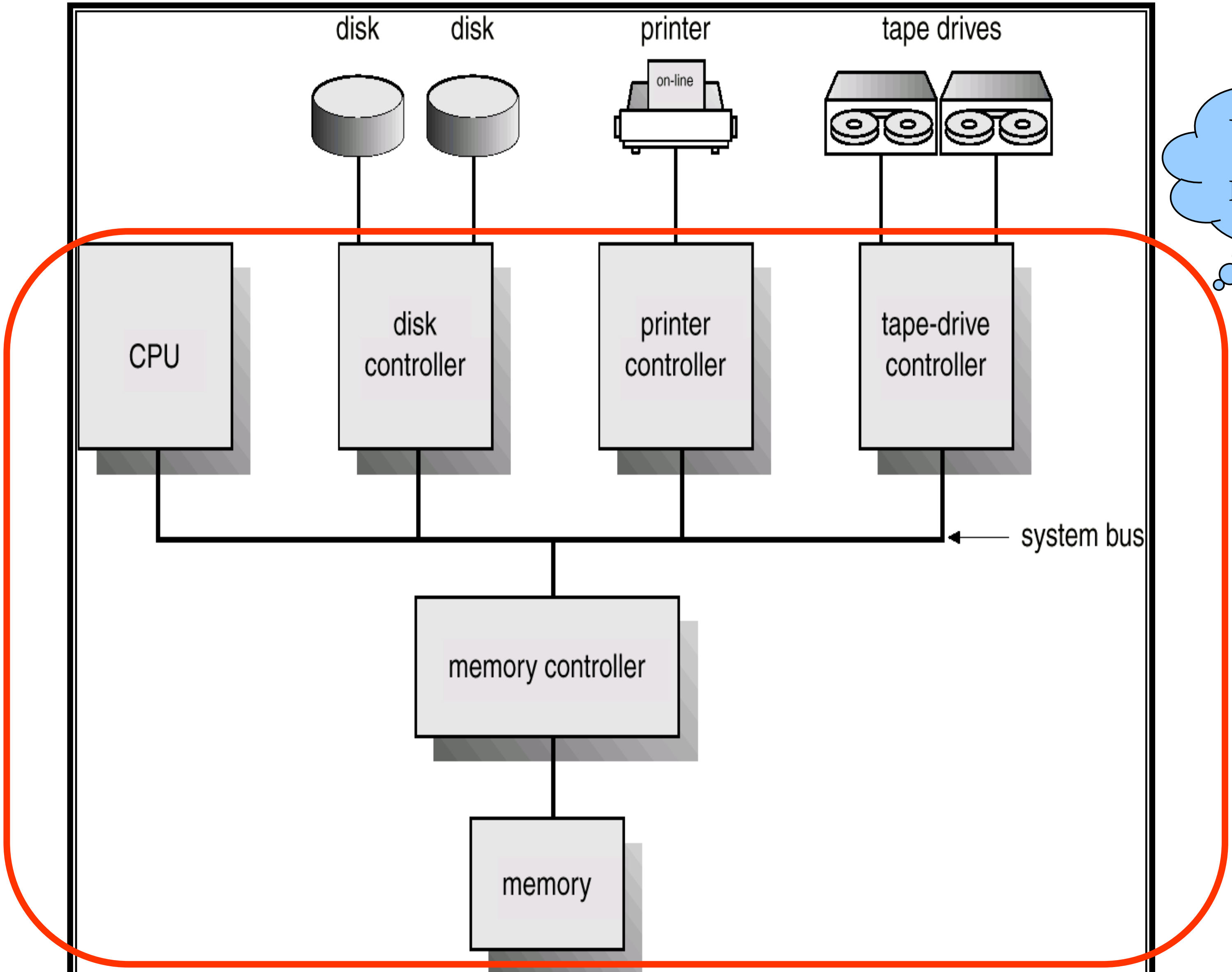
1.4 Evolution of Operating Systems

1.5 Personal Computing, Distributed Computing and Client/Server Computing

1.6 Machine Languages, Assembly Languages and High-level Languages

1.8 C#

Computer-system structures



Upon mainboard

Mainboards

- **Printed Circuit Board (PCB)**
  - **Hardware component that provides electrical connections between devices**
- **The mainboard is the central PCB in a system**
  - **Devices such as processors and main memory are attached**
  - **Include chips to perform low-level operations (e.g., BIOS, video card)**

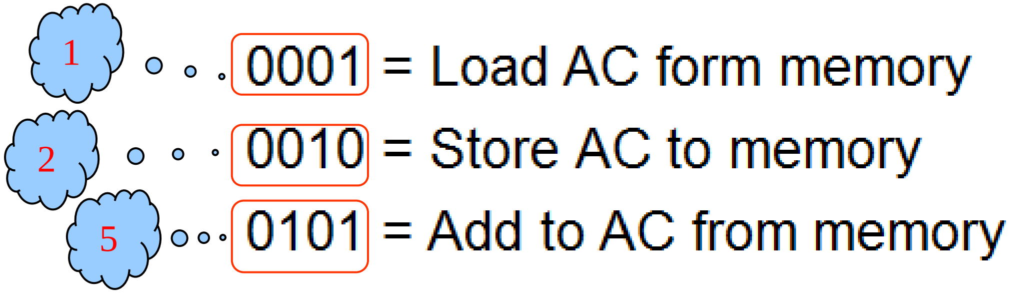
Processors (CPU)

- **A processor is hardware that executes machine-language**
  - **CPU executes the instructions of a program**
- **Coprocessor executes special-purpose instructions**
  - **Ex., graphics or video coprocessors**

◀ ▶

Machine Language and Assembly Language

- **Machine language**
  - **Defined by the computer’s hardware design**
  - **Consists of streams of numbers (1s and 0s) that instruct computers how to perform elementary operations**
  - **A computer can understand only its own machine language**
  - **Ex: for a set of machine codes for a CPU**



0001 = Load AC form memory

0010 = Store AC to memory

0101 = Add to AC from memory

ps: AC: accumulator

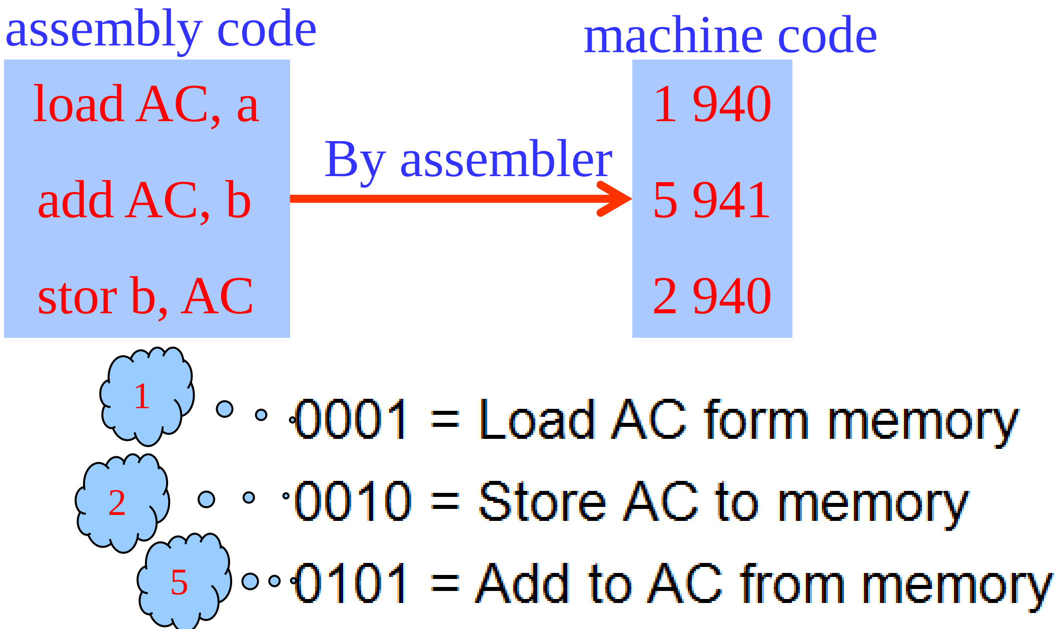
◀ ▶

Machine Language and Assembly Language

- **Assembly language**
  - **Represents machine-language instructions using English-like abbreviations**
  - **Assemblers convert assembly language to machine language**
  - **Ex: assumption:**

    a is in memory address 940,

    b is in memory address 941



assembly code

load AC, a

add AC, b

stor b, AC

By assembler

machine code

1 940

5 941

2 940

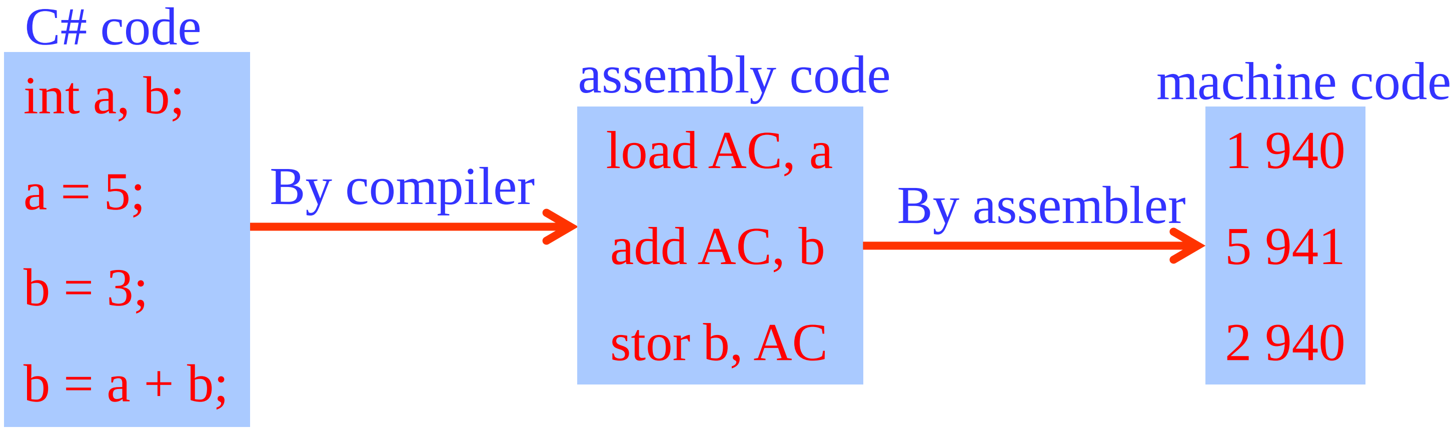
0001 = Load AC form memory

0010 = Store AC to memory

0101 = Add to AC from memory

Machine Language and Assembly Language

- **High-level languages**
  - **Instructions look similar to everyday English**
  - **Accomplish more substantial tasks with fewer statements**
  - **Require compilers or interpreters**
  - **Ex:**



C# code

```
int a, b;
a = 5;
b = 3;
b = a + b;
```

By compiler

assembly code

```
load AC, a
add AC, b
stor b, AC
```

By assembler

machine code

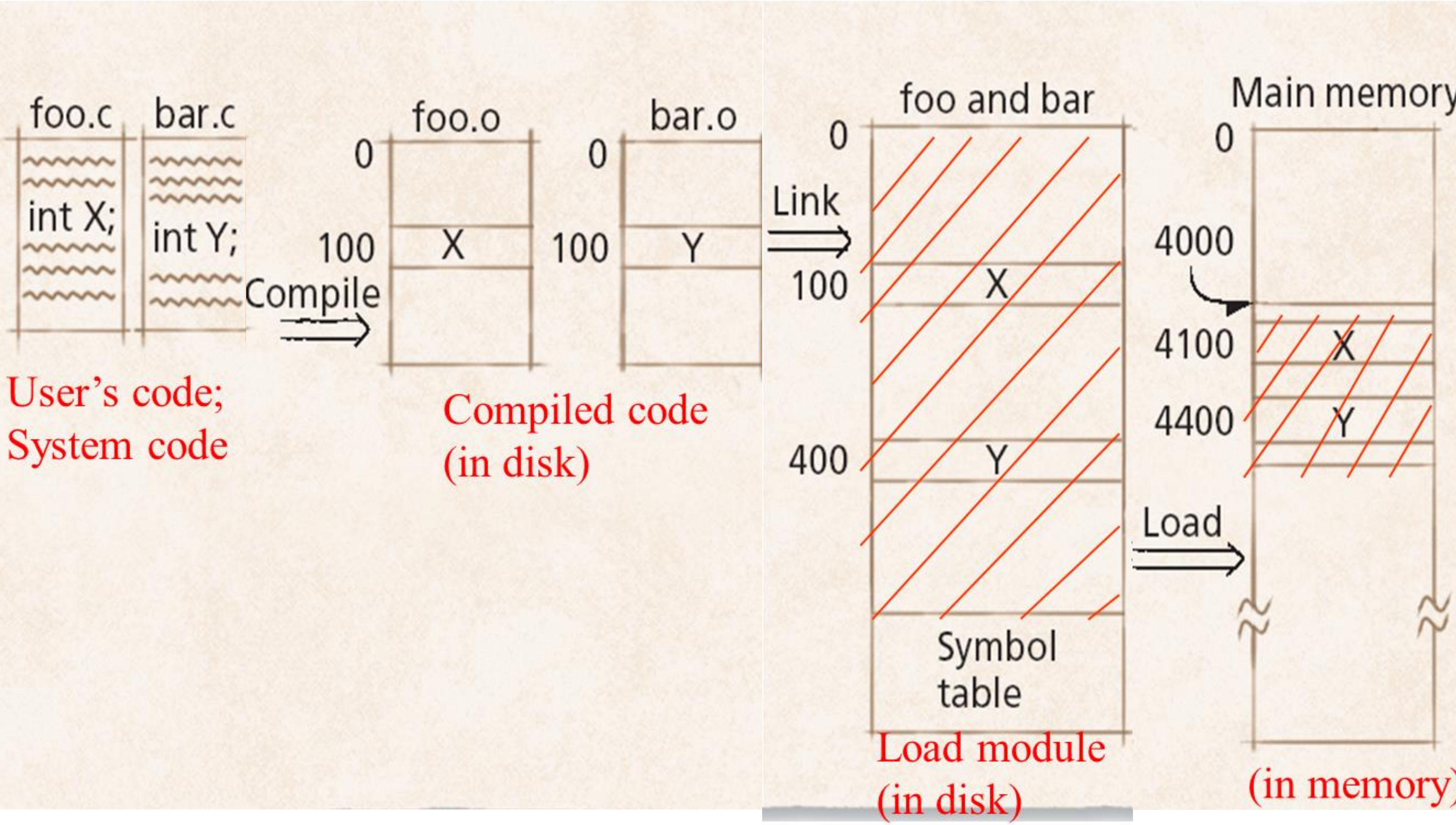
```
1 940
5 941
2 940
```

Compiling, Linking and Loading

- **Before a high-level-language program can execute, it must be:**
  - **Translated into machine language**
  - **Linked with various other machine-language programs on which it depends**
  - **Loaded into memory**

◀ ▶

Compiling, linking and loading



Linking & Loading

- **Linkers**
  - **Integrate precompiled modules called libraries referenced by a program**
  - **Produce an integrated module called a load module**
- **Loaders**
  - **Convert relative addresses to physical addresses**
  - **Place each instruction and data unit in main memory**

◀ ▶

1.8 C#

- C#
  - Developed at Microsoft by a team led by Anders Hejlsberg and Scott Wiltamuth
  - Event driven, object oriented, visual programming language
  - Based from C, C++ and Java
  - Incorporated into .NET platform
    - Web based applications can be distributed
      - mobile devices and desktop computers
    - Programs that can be accessed by anyone through any device
    - Allows communicating with different computer languages
  - **I**ntegrated **D**esign **E**nvironment (IDE)
    - Makes programming and debugging fast and easy
    - **R**apid **A**pplication **D**evelopment (RAD)

◀ ▶

1.16 Introduction to Microsoft .NET

- .NET initiative
  - Introduced by Microsoft (June 2000)
    - Vision for embracing the Internet in software development
  - Independence from specific language or platform
    - Applications developed in any .NET compatible language
      - Visual Basic .NET, Visual C++ .NET, C# and more
    - Programmers can contribute to applications using the language in which they are most competent
  - Architecture capable of existing on multiple platforms

◀ ▶