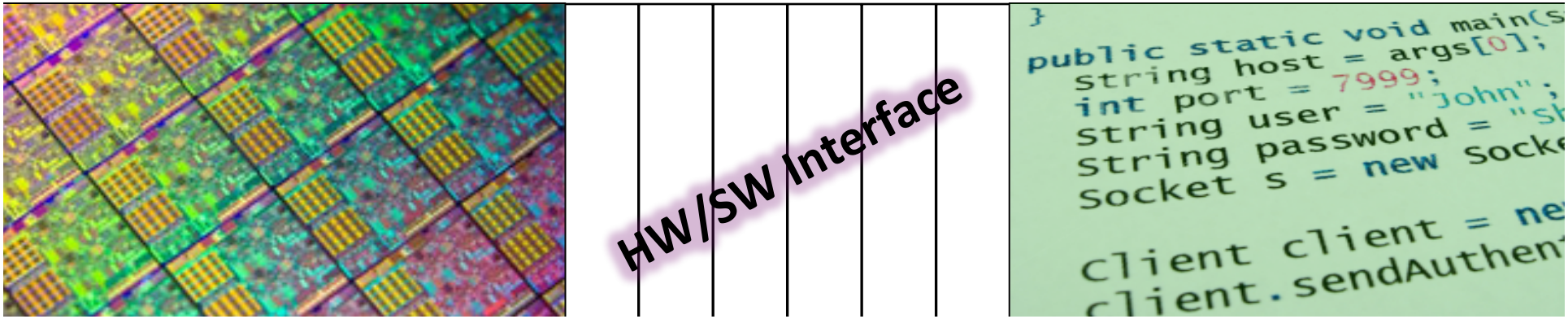
What is this class about?

- What is hardware? software?
- What is a hardware/software interface?
- Why do we need to understand this interface?



C/Java, assembly, and machine code

```
if (x != 0) y = (y+z)/x;
```

```
cmpl    $0, -4(%ebp)
je      .L2
movl    -12(%ebp), %eax
movl    -8(%ebp), %edx
leal    (%edx, %eax), %eax
movl    %eax, %edx
sarl    $31, %edx
idivl   -4(%ebp)
movl    %eax, -8(%ebp)
```

```
10000011011111000010010000011100000000000
0111010000011000
10001011010001000010010000010100
10001011010001100010010100010100
10001101000001000000010
1000100111000010
11000001111110100011111
11110111011111000010010000011100
10001001010001000010010000011000
```

C/Java, assembly, and machine code

```
if (x != 0) y = (y+z)/x;
```



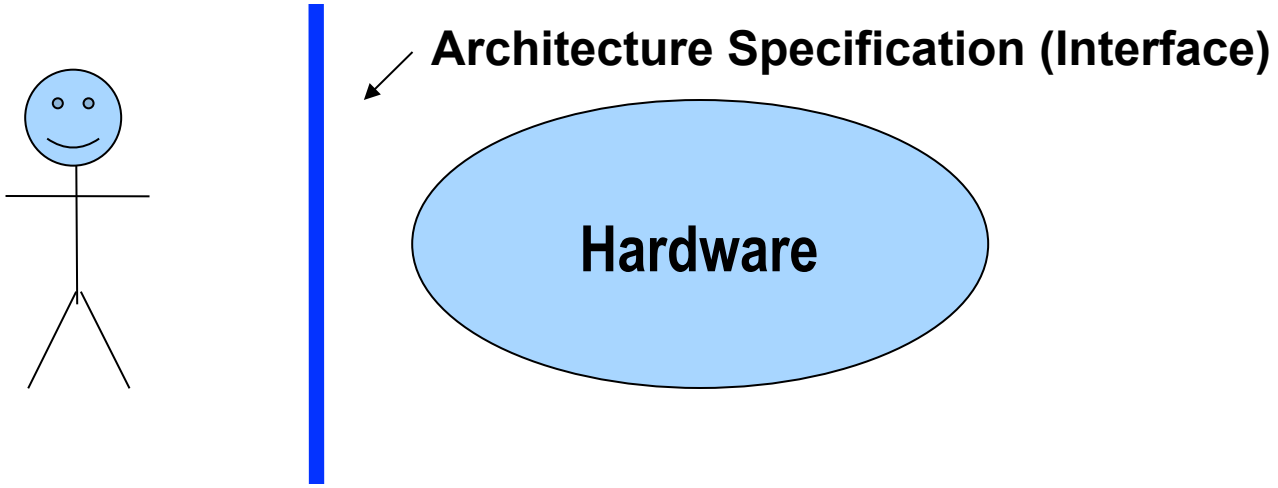
```
cmpl    $0, -4(%ebp)
je      .L2
movl    -12(%ebp), %eax
movl    -8(%ebp), %edx
leal    (%edx, %eax), %eax
movl    %eax, %edx
sarl    $31, %edx
idivl   -4(%ebp)
movl    %eax, -8(%ebp)
```

```
10000011011111000010010000011100000000000
0111010000011000
10001011010001000010010000010100
10001011010001100010010100010100
10001101000001000000010
1000100111000010
11000001111110100011111
11110111011111000010010000011100
10001001010001000010010000011000
```

- **The three program fragments are equivalent**
- **You'd would rather write C! – a more human-friendly language**
- **The hardware likes bit strings! – 0 and 1 as low or high voltages**
  - The machine instructions are actually much shorter than the number of bits we would need to represent the characters in the assembly language

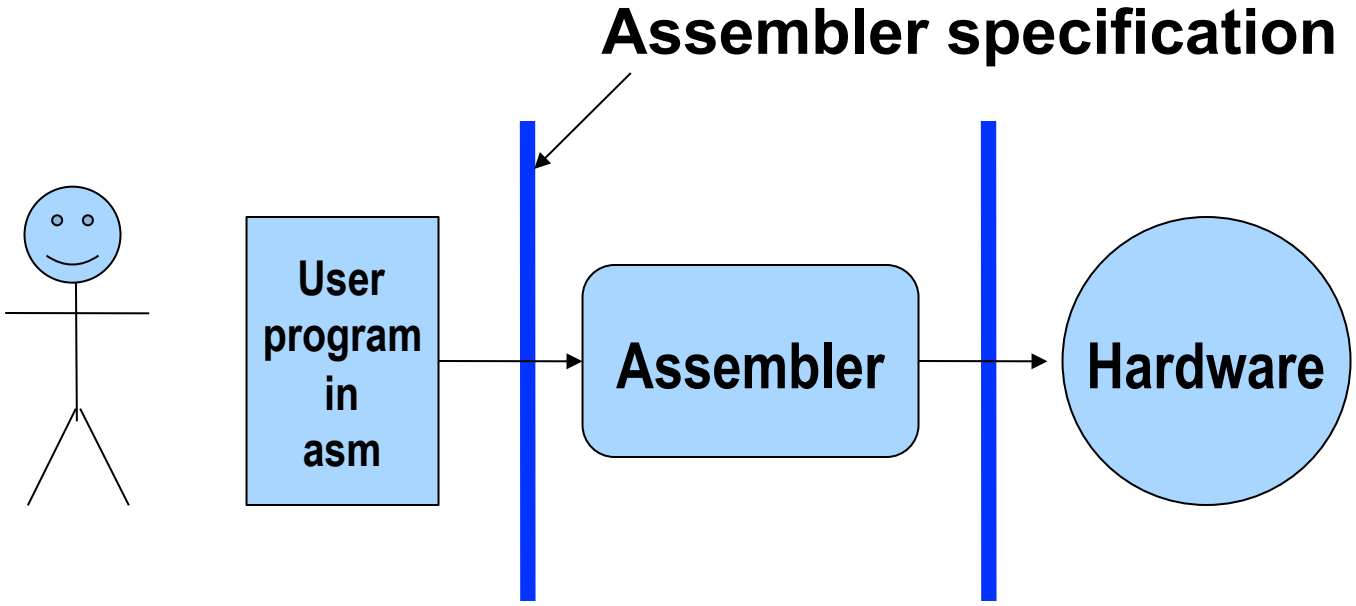
HW/SW Interface: The Historical Perspective

- **Hardware started out quite primitive**
  - Hardware designs were expensive ⇒ instructions had to be very simple – e.g., a single instruction for adding two integers
- **Software was also very primitive**
  - Software primitives reflected the hardware pretty closely



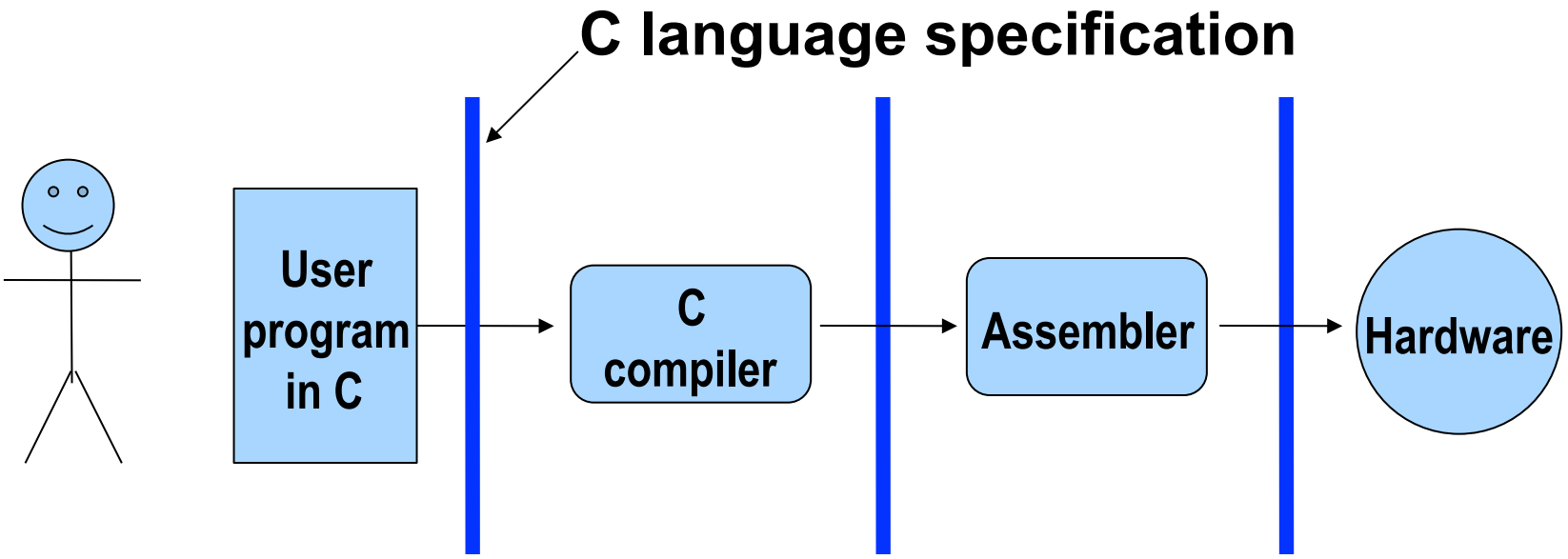
HW/SW Interface: Assemblers

- **Life was made a lot better by assemblers**
  - 1 assembly instruction = 1 machine instruction, but...
  - different syntax: assembly instructions are character strings, not bit strings, a lot easier to read/write by humans
  - can use symbolic names

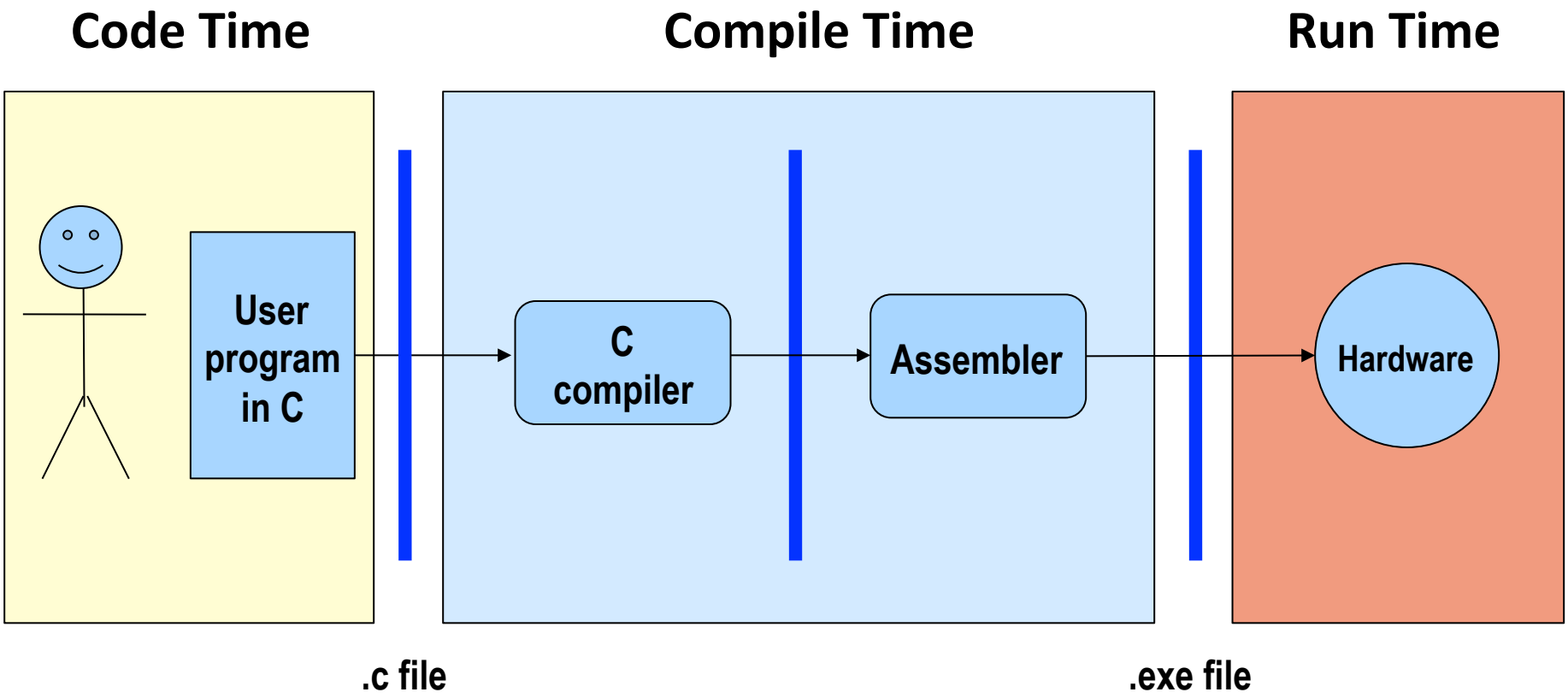


HW/SW Interface: Higher-Level Languages

- **Higher level of abstraction:**
  - 1 line of a high-level language is compiled into many (sometimes very many) lines of assembly language



HW/SW Interface: Code / Compile / Run Times



Note: The compiler and assembler are just programs, developed using this same process.

The Big Theme

- **THE HARDWARE/SOFTWARE INTERFACE**

- **How does the hardware (0s and 1s, processor executing instructions) relate to the software (Java programs)?**
- **Computing is about abstractions (but we can't forget reality)**
- **What are the abstractions that we use?**
- **What do YOU need to know about them?**
  - When do they break down and you have to peek under the hood?
  - What bugs can they cause and how do you find them?
- **Become a better programmer and begin to understand the important concepts that have evolved in building ever more complex computer systems**