Software Engineering

Lesson #06 - Lecture

Lesson #06 - Lecture

Your KBTU 202309 Software Engineering
class information is updating …

Lesson #06 update is in progress

This will take around 2 hours to complete

Please, don’t turn off your head

Design and implementation

Design and implementation

Design and implementation

```
#include<iostream>
Using namespace std;

int main()
{
      cout << “Design and implementation” << endl;

      return 0;
}
```

Agenda: Lesson #06 - Software Engineering - Lecture

1. Object-oriented design using UML
2. Design patterns
3. Implementation issues
4. Open-source development

Agenda: Lesson #06 - Software Engineering - Lecture

Object-oriented design using UML

An object-oriented system is made up of interacting objects that maintain their own local state and provide operations on that state

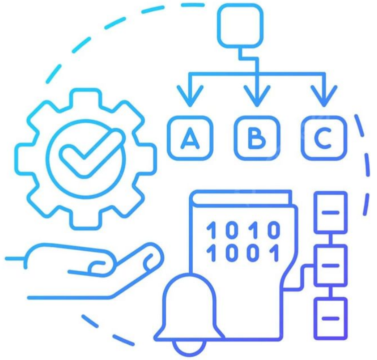
The representation of the state is private and cannot be accessed directly from outside the object

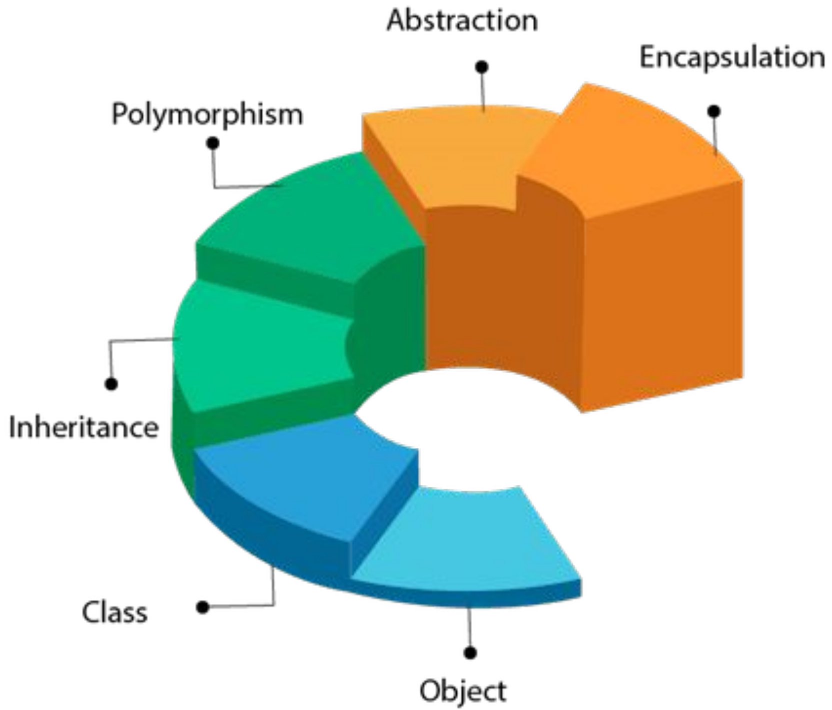
Object-oriented design using UML

Objects include both data and operations to manipulate that data

They may therefore be understood and modified as stand-alone entities

Object-oriented design using UML





Object-oriented design using UML

System context and interactions

- The first stage in any software design process is to develop an understanding of the relationships between the software that is being designed and its external environment

Object-oriented design using UML

Architectural design

- Once the interactions between the software system and the system's environment have been defined, you use this information as a basis for designing the system architecture

- Of course, you need to combine this knowledge with your general knowledge of the principles of architectural design and with more detailed domain knowledge

Object-oriented design using UML

Object class identification

- By this stage in the design process, you should have some ideas about the essential objects in the system that you are designing

- As your understanding of the design develops, you refine these ideas about the system objects

Object-oriented design using UML

Design models

- Design or system models show the objects or object classes in a system
- They also show the associations and relationships between these entities
- These models are the bridge between the system requirements and the implementation of a system

Object-oriented design using UML

Interface specification

- An important part of any design process is the specification of the interfaces between the components in the design
- You need to specify interfaces so that objects and subsystems can be designed in parallel
- Once an interface has been specified, the developers of other objects may assume that interface will be implemented

Agenda: Lesson #06 - Software Engineering - Lecture

1. Object-oriented design using UML
2. **Design patterns**
3. Implementation issues
4. Open-source development

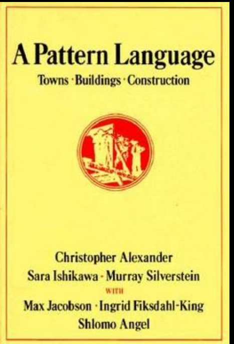
Design Patterns

Design patterns were derived from ideas put forward by Christopher Alexander (Alexander 1979), who suggested that there were certain common patterns of building design that were inherently pleasing and effective

Design patterns

Design Patterns

In software, Alexander is regarded as the father of the pattern language movement



A Pattern Language

Towns · Buildings · Construction

Christopher Alexander
Sara Ishikawa · Murray Silverstein
WITH
Max Jacobson · Ingrid Fiksdahl-King
Shlomo Angel

Design Patterns

Patterns and Pattern Languages are ways to describe best practices, good designs, and capture experience in a way that it is possible for others to reuse this experience

Design Patterns

Software Design Patterns and Principles (quick overview)

https://www.youtube.com/watch?v=WV2Ed1QTst8&t=453s

Design Patterns

What are Design Patterns and Should You Learn Them?

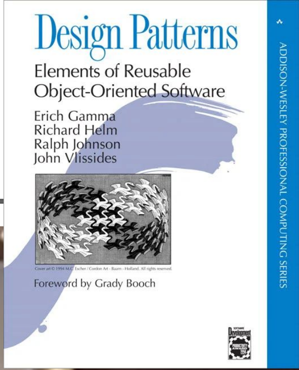
https://www.youtube.com/watch?v=Hz1GGUmS7a4

Design Patterns

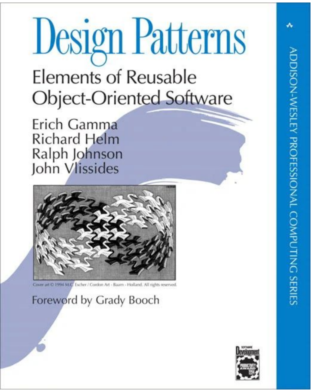
What Are Design Patterns?

https://www.youtube.com/watch?v=BWprw8UHIzA

Design patterns

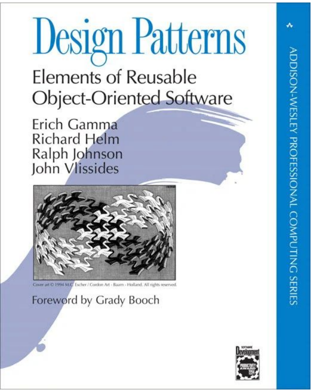

Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

THE 23 GANG OF FOUR DESIGN PATTERNS


Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

C	Abstract Factory
S	Adapter
S	Bridge
C	Builder
B	Chain of Responsibility
B	Command
S	Composite
S	Decorator
S	Facade
C	Factory Method
S	Flyweight
B	Interpreter
B	Iterator
B	Mediator
B	Memento
C	Prototype
S	Proxy
B	Observer
C	Singleton
B	State
B	Strategy
B	Template Method
B	Visitor

Gang of Four (GoF) patterns


Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

- **Creational Patterns** *(abstracting the object-instantiation process)*

Factory Method	Abstract Factory	Singleton
Builder	Prototype	

- **Structural Patterns** *(how objects/classes can be combined)*

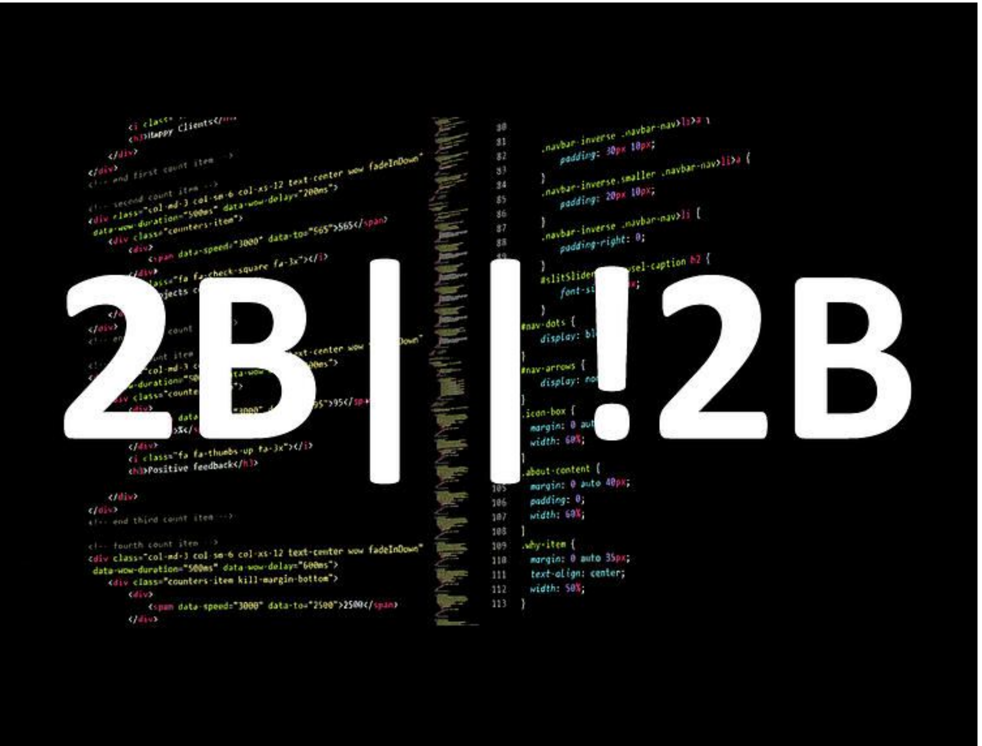
Adapter	Bridge	Composite
Decorator	Facade	Flyweight
Proxy		

- **Behavioral Patterns** *(communication between objects)*

Command	Interpreter	Iterator
Mediator	Observer	State
Strategy	Chain of Responsibility	Visitor
Template Method		

Agenda: Lesson #06 - Software Engineering - Lecture

Implementation issues



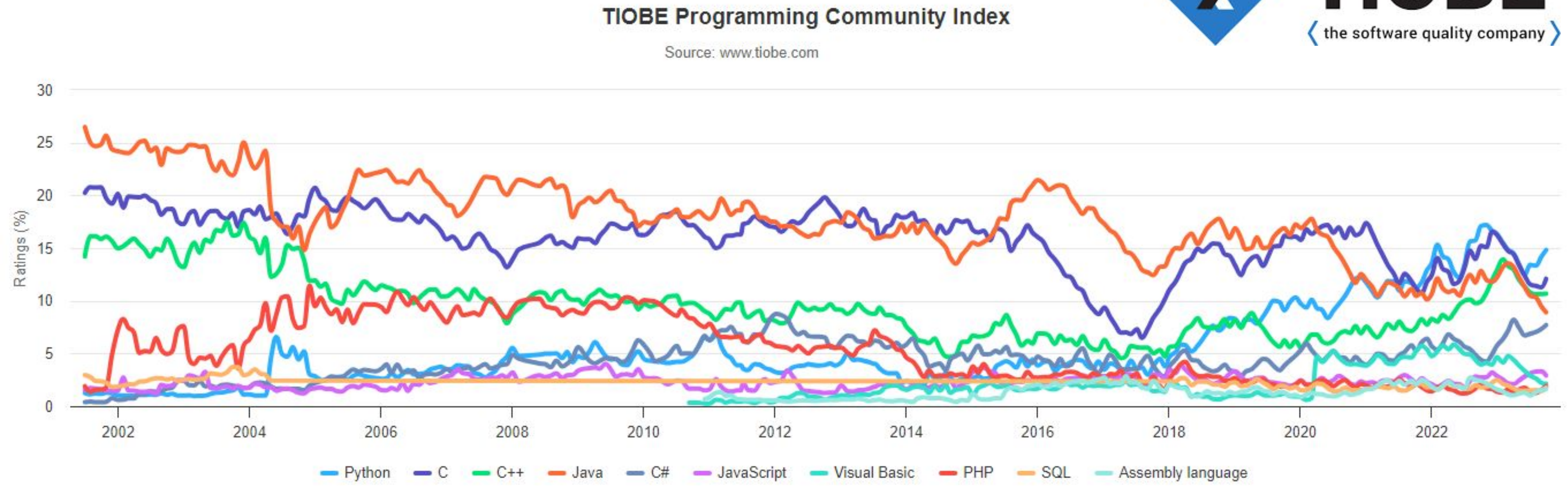
Implementation issues

A critical stage of this process is, of course, system implementation, where you create an executable version of the software

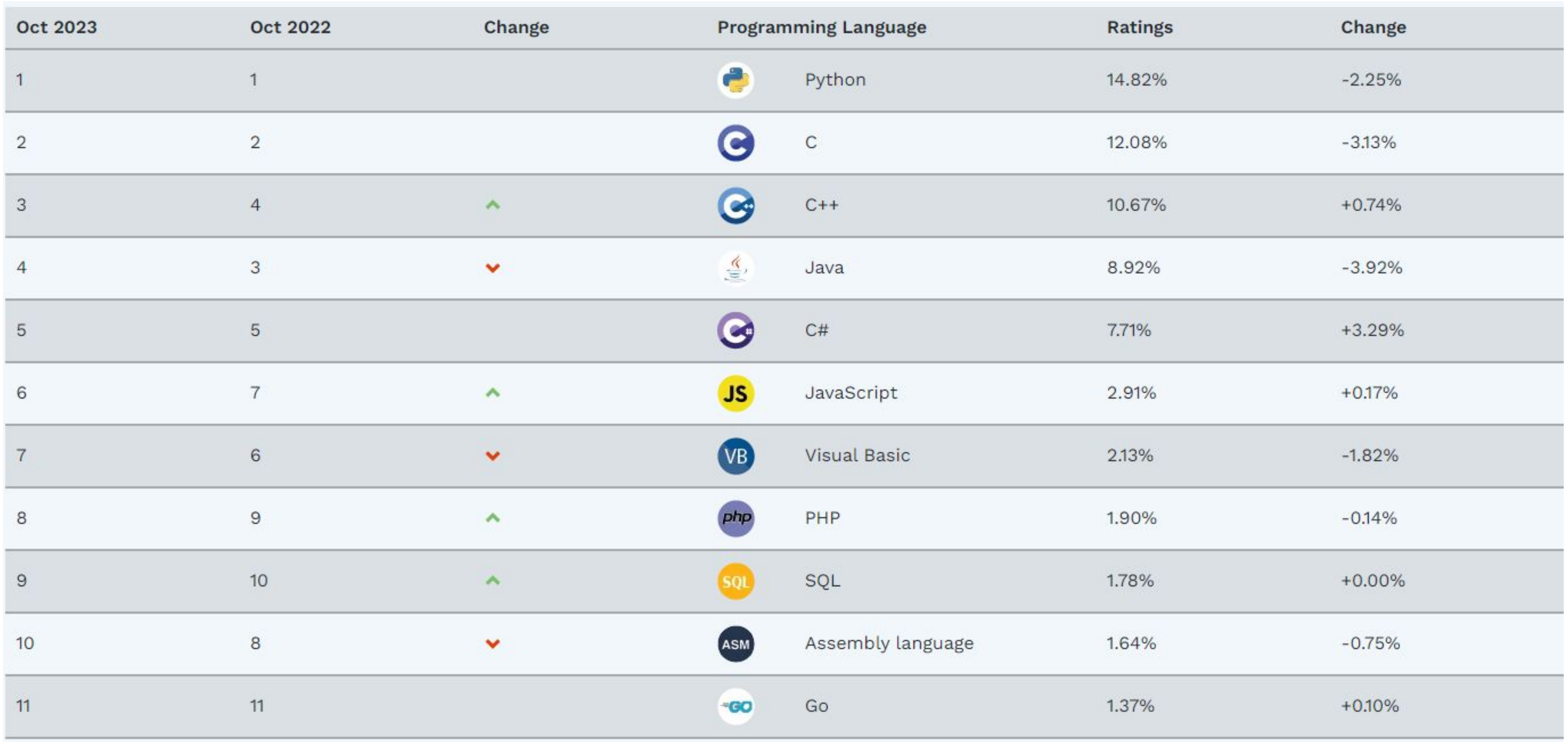
Implementation may involve developing programs in high- or low-level programming languages or tailoring and adapting generic, off-the-shelf systems to meet the specific requirements of an organization

Implementation issues

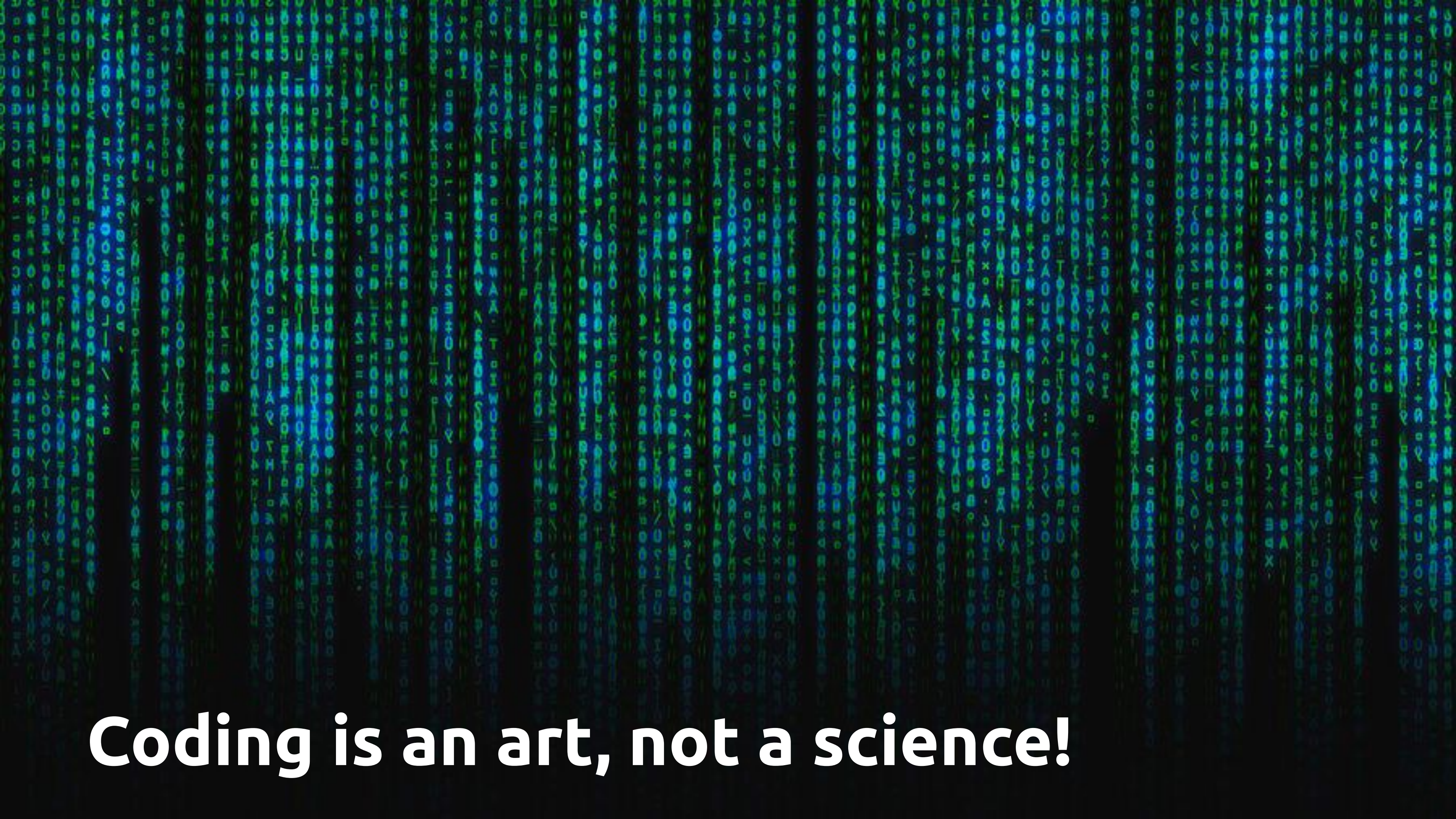
TOIBE October 2023



Implementation issues

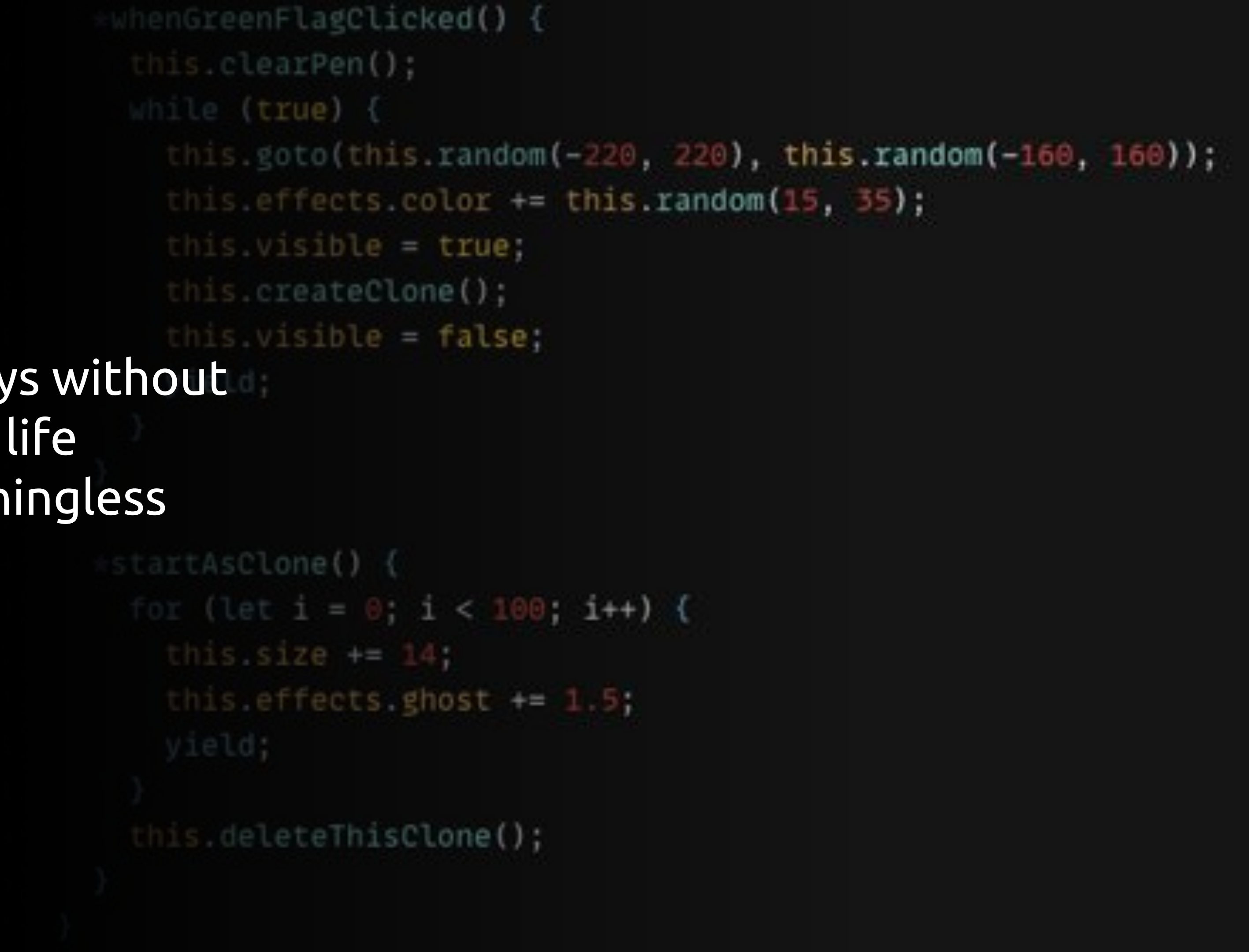


Oct 2023	Oct 2022	Change	Programming Language	Ratings	Change
1	1		Python	14.82%	-2.25%
2	2		C	12.08%	-3.13%
3	4	▲	C++	10.67%	+0.74%
4	3	▼	Java	8.92%	-3.92%
5	5		C#	7.71%	+3.29%
6	7	▲	JavaScript	2.91%	+0.17%
7	6	▼	Visual Basic	2.13%	-1.82%
8	9	▲	PHP	1.90%	-0.14%
9	10	▲	SQL	1.78%	+0.00%
10	8	▼	Assembly language	1.64%	-0.75%
11	11		Go	1.37%	+0.10%



Coding is an art, not a science!

After three days without programming, life becomes meaningless



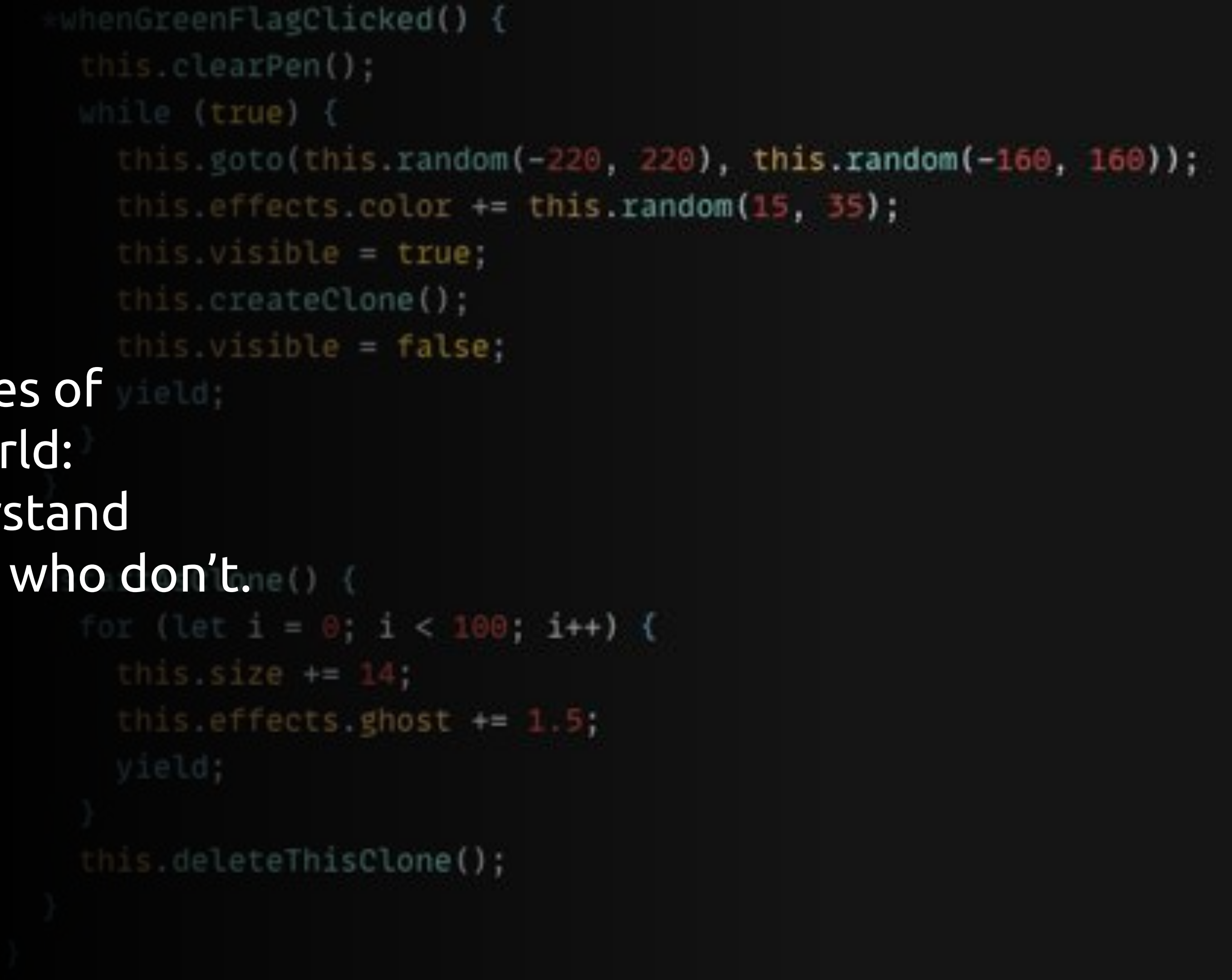
```
*whenGreenFlagClicked() {
  this.clearPen();
  while (true) {
    this.goto(this.random(-220, 220), this.random(-160, 160));
    this.effects.color += this.random(15, 35);
    this.visible = true;
    this.createClone();
    this.visible = false;
    yield;
  }
}

*startAsClone() {
  for (let i = 0; i < 100; i++) {
    this.size += 14;
    this.effects.ghost += 1.5;
    yield;
  }
  this.deleteThisClone();
}
```

```
    return false;
  }
}
var marker = new toogle.secure.Marker({image: log, position:
    marker.addListener('click', function() {
    infowindow.open(log, marker);
            }      </script>script>script>
    });
<form name="myForm" action="/action_page.php" "return validate()" method="post">
    Name: <input type="text" name="fname">
    <input type="submit" value="Submit">
    </form>form>
    <script syncronized user interface
    src="https://url.password/login/api/js?key="></script>script>script>
    </body>body>body>                    secure access guaranteed.
          html>;
```

First solve the problem, then write code!

There are 10 types of
people in the world:
those who understand
binary and those who don’t.



```
*whenGreenFlagClicked() {
  this.clearPen();
  while (true) {
    this.goto(this.random(-220, 220), this.random(-160, 160));
    this.effects.color += this.random(15, 35);
    this.visible = true;
    this.createClone();
    this.visible = false;
    yield;
  }
}

        ne() {
  for (let i = 0; i < 100; i++) {
    this.size += 14;
    this.effects.ghost += 1.5;
    yield;
  }
  this.deleteThisClone();
}
}
```

Algorithm *(noun.)*

Word used by programmers when...
they do not want to explain what they did.

Programmer *(noun.)*

A machine that turns coffee into code.

The programmer's wife
sent him to the grocery store.

Her instructions were:

**"Buy butter. See if
they have eggs. If they do,
buy 10"**

So he bought 10.

A foo walks into a bar,
takes a look around
and says "**Hello World!**".

Implementation issues

Reuse

- From the 1960s to the 1990s, most new software was developed from scratch, by writing all code in a high-level programming language

- The only significant reuse or software was the reuse of functions and objects in programming language libraries

Implementation issues

Configuration management

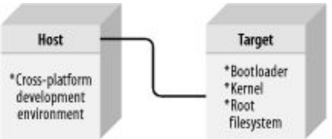
- In software development, change happens all the time, so change management is absolutely essential

- When several people are involved in developing a software system, you have to make sure that team members don’t interfere with each other’s work

Implementation issues

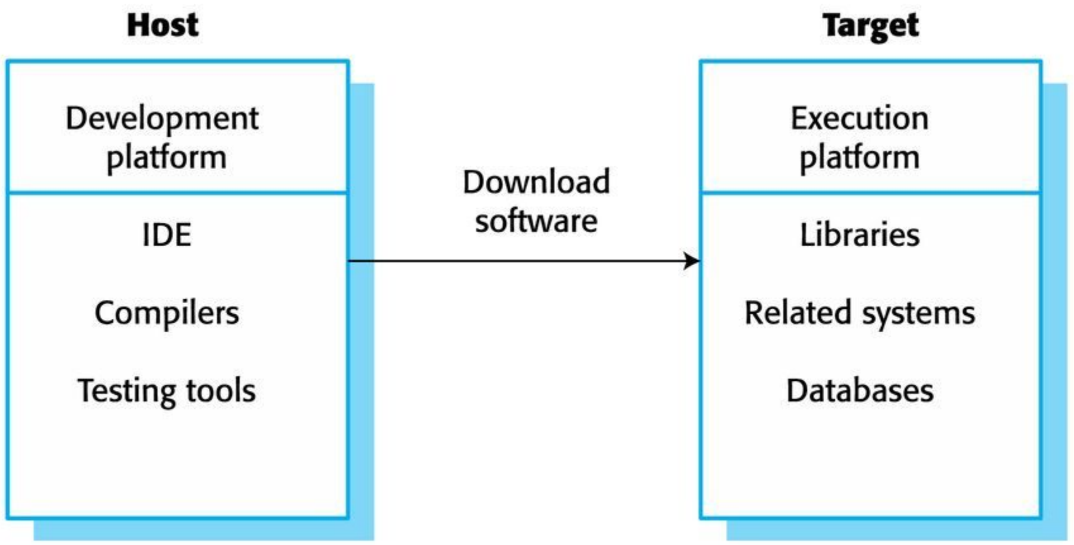
Host-target development

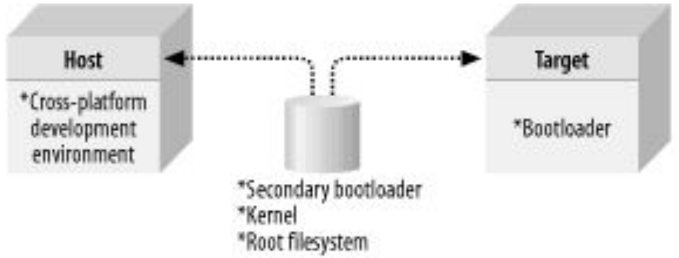
- Most professional software development is based on a host-target model
- Software is developed on one computer (the host) but runs on a separate machine (the target)
- More generally, we can talk about a development platform (host) and an execution platform (target)

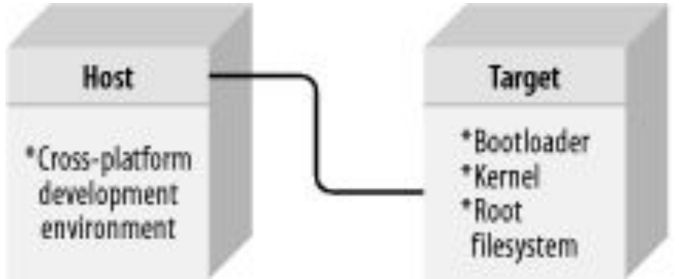


Implementation issues

Host-target development







Implementation issues

Host-target development

Agenda: Lesson #06 - Software Engineering - Lecture

1. Object-oriented design using UML
2. Design patterns
3. Implementation issues
4. **Open-source development**

Open-source development

Open-source development is an approach to software development in which the source code of a software system is published and volunteers are invited to participate in the development process

Open-source development

Open-source software extended this idea by using the Internet to recruit a much larger population of volunteer developers

Open-source development

Open-source software is the backbone of the Internet and SE

Open-source development

The Linux operating system is the most widely used server system, as is the open-source Apache web server

Open-source development

Other important and universally used open-source products are Java, the Eclipse IDE, and the mySQL database management system

Open-source development

The Android operating system is installed on millions of mobile devices

Open-source development

Open-source licensing

- Although a fundamental principle of open-source development is that source code should be freely available, this does not mean that anyone can do as they wish with that code

- Legally, the developer of the code (either a company or an individual) owns the code

Open-source development

The Rise Of Open-Source Software

https://www.youtube.com/watch?v=SpeDK1TPbew&t=685s

Agenda: Lesson #06 - Software Engineering - Lecture

Q & A