# Abstraction and Encapsulation Software Development

## CSc3350

Dr. William Greg Johnson
Department of Computer Science
Georgia State University

Georgia State University | COLLEGE OF ARTS & SCIENCES

## Abstraction and Encapsulation

- Differences
- Importance
- Benefits
- Why do these things?

# Abstraction vs. Encapsulation

| ABSTRACTION | ENCAPSULATION |
|---|---|
| Abstraction is the process or method of gaining the information. | While encapsulation is the process or method to contain the information. |
| In abstraction, problems are solved at the design or interface level. | While in encapsulation, problems are solved at the implementation level. |
| Abstraction is the method of hiding the unwanted information. | Whereas encapsulation is a method to hide the data in a single entity or unit along with a method to protect information from outside. |
| We can implement abstraction using abstract class and interfaces. | Whereas encapsulation can be implemented using by access modifier i.e. private, protected and public. |
| In abstraction, implementation complexities are hidden using abstract classes and interfaces. | While in encapsulation, the data is hidden using methods of getters and setters. |
| The objects that help to perform abstraction are encapsulated. | Whereas the objects that result in encapsulation need not be abstracted. |

## Why is encapsulation important?

- One of the foundation pillars of OOP
- Limits the access and visibility of variables and methods
- Keeps local methods unknown to users of the class through declaring private
- If private variable needs to be changed in the class, keep it hidden with private and allow public accessor and mutator methods
- Allows flexibility, Reusability, and Maintainability

## Benefit of encapsulation:

- The fields of a class can be made read-only or write-only.
- A class can have total control over what is stored in its fields.
- The users of a class do not know how the class stores its data. A class can change the data type of a field and users of the class do not need to change any of their code.

## Why do encapsulation?

- The purpose of encapsulation is to achieve potential for change: the internal mechanisms of the component can be improved without impact on other components, or the component can be replaced with a different one that supports the same public interface.
- Encapsulation also protects the integrity of the component, by preventing users from setting the internal data of the component into an invalid or inconsistent state.

## Why do encapsulation?

- It reduces system complexity and thus increases robustness, by limiting the interdependencies between software components.
- Encapsulation = good cohesion

## Best practices with encapsulation

- Make private members as private as possible. Only make members of a class public if they need to be accessed by other code in the application that instantiated the class.
- Use getters and setters to access and modify all private variables that need initialization and modifications.
- Document your getters and setters to explain what they do and how to use them. (Black Box Syndrome: loose site of external behavior, focus on internal operations.)
- Use inheritance to share encapsulated code because it allows you to reuse code in other classes without duplication.

Georgia State University | COLLEGE OF ARTS & SCIENCES

# What is abstraction?

- Where a programmer hides all but the relevant data about an object in order to reduce complexity and increase efficiency.
- Abstraction is essential in the construction of programs. It places the emphasis on what an object is or does rather than how it is represented or how it works. Thus, it is the primary means of managing complexity in large programs.

## Abstraction in data

- An abstract data type is defined only by the type of data stored and the operations that may be performed on concrete instances of the type.
- Developers access concrete instances of an ADT using only its operations, and they are unaware of how the operations are implemented within the internal structure of the data type.

## Abstraction in behavior

Methods and classes

- Abstract methods are methods with no body specification. Subclasses must provide the method statements for their meaning.
- If the method was one provided by the superclass, it would require overriding in each subclass.
- And if one forgot to override, the applied method statements may be inappropriate.

# Best practices with abstraction

- Only hide the implementation details that are necessary and do not hide ones that are important for users to understand or that may need to be changed in the future.
- Use clear and concise names for abstract classes, interfaces, and methods to make it easier for developers to understand how to use them.
- Document your abstractions to explain what they do and how to use them.
- Use test-driven development to test your abstractions to ensure that your they are implemented correctly and that they meet the needs of users.

## What is the biggest difference?

- Abstraction is about hiding what something does, while encapsulation is about hiding how something does it.

Examples:

1. A bank account has a balance, and it can be deposited into or withdrawn from. However, you cannot directly access the bank account's balance. You must use the bank account's methods to deposit into or withdraw from the account. This is ***encapsulation***. The bank account's balance is encapsulated from the outside world.
2. A car with a GPS system. is encapsulated, so the driver doesn't need to know how it works in order to use it. The GPS system itself is an example of ***abstraction***. The GPS system hides the complexity of how it navigates from the driver. The driver only needs to know the location, and the GPS system will take care of getting them to a destination location.

## Summary

Both are powerful techniques and share characteristics that can be used to improve the quality of software and its development.

- Abstraction helps to make code more readable, maintainable, reusable, and secure.
- Encapsulation helps to organize code, reduce errors, increase reusability, and improve data security.

## Attributions

- https://www.scribd.com/presentation/54236491/T3-Object-Oriented-Programming-in-C
- https://www.scribd.com/presentation/605253153/03-Encapsulation
- https://www.scribd.com/presentation/472538481/Abstraction-Encapsulation-in-C