Data Structures & Algorithms

Object-oriented
Programming - Principles

OOP Principles

- Abstraction
  - Hiding the inner workings
- Encapsulation
  - Binding data and behavior
- Modularity
  - Separation of functionality into modules

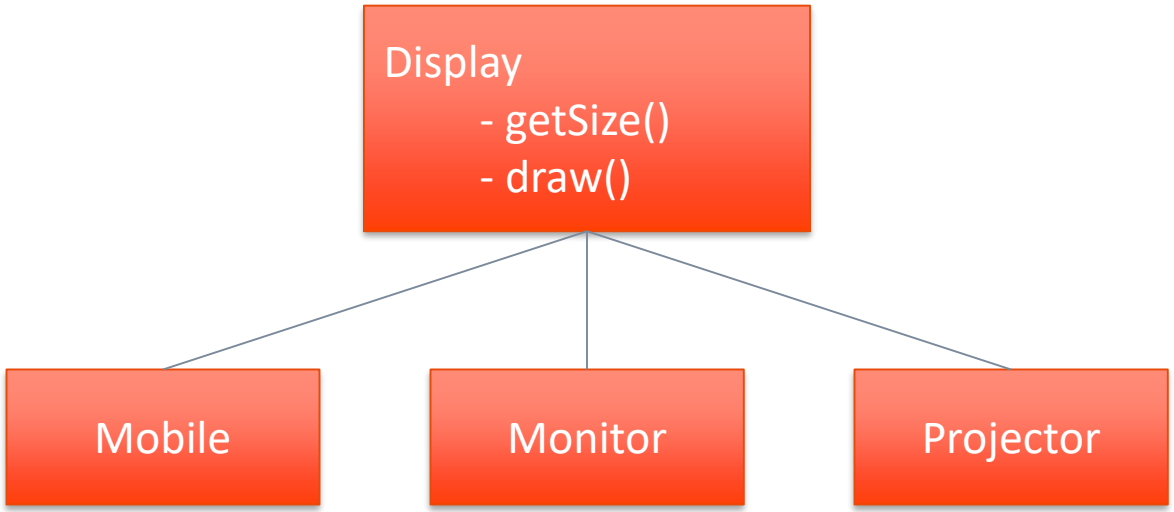
OOP Principles

- **Abstraction:** Distill a complicated system to its most fundamental parts and describe these in simple, but precise language.

- More transparent design
- Easier to predict behavior
- Describe most essential things

OOP Principles - Abstraction

- **Example:**
  - App:
    - Just needs to know that there is a display, but not which type of display



OOP Principles

- **Encapsulation:** Components should hide internal details for other components
  - Hide implementation details
  - Hide data fields

- Possible to change later the details

OOP Principles -Encapsulation

- Encapsulation benefits the **adaptability**
  - implementation details can change without affecting other parts

```
private int balance = 100;

public int checkBalance() {
      return balance;
}
```

OOP Principles

- **Modularity:** The different components of a system should be divided into separate and independent functional units

- Enhances the **reusability** of the program code
  - modules can easily be reused in other programs

OOP Principles - Modularity

- **Example:**

```
void DrawRectangle(x,y,width, height)
        DrawLine(x,y,x+width,y)
        DrawLine(x,y,x,y+height)
        DrawLine(x+width,y,x+width,y+height)
        DrawLine(x,y+height,x+width,y+height)
```