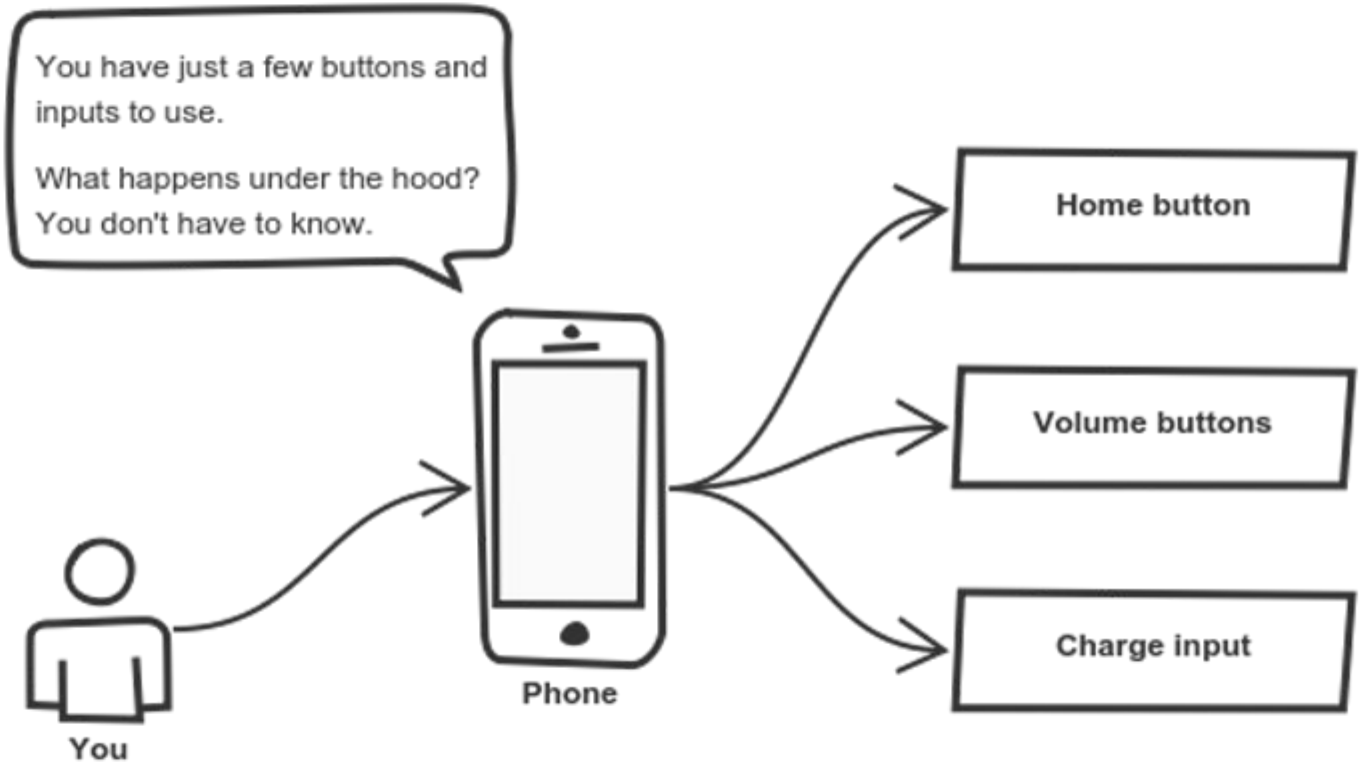
# Object-oriented Programming

## Week 1 | Lecture 2

# Recap



You have just a few buttons and inputs to use.

What happens under the hood? You don't have to know.

# Information Hiding

- Implementation details are hidden from the outside world

- Information is stored within the object

- It can only be *changed* by the object itself

# **Example:** Smartphone

- All logic is hidden inside the handset
- You use HOME button to get to the home screen without knowing "how" it was done

# Advantages of Information Hiding

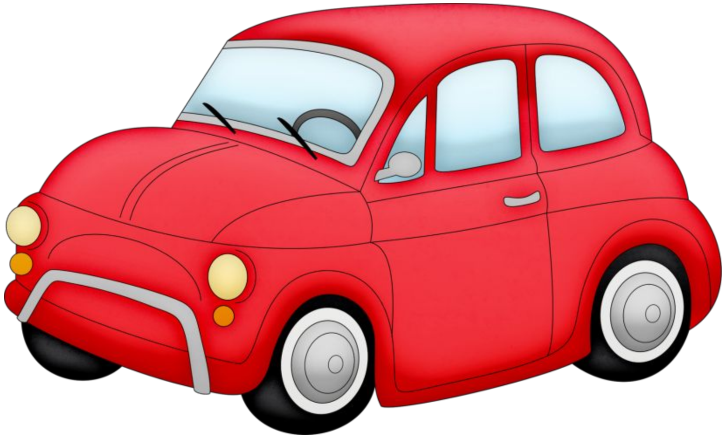
- Simplifies the model by hiding implementation details

- Prevents accidental access

- Prevents illegal access or manipulation

# Classes

- Accelerator Pedal
- Brake Pedal
- Steering Wheel



- User-friendly “interfaces” to control the car
- Their mechanism is *housed* inside the engineering drawings or blueprints

# Classes

```
class Car
{
        void accelerate()
    { \\ logic for acceleration }

        void brake()
    { \\ logic for brakes }
};
```

# Classes

```
class Car
{
        string model;
    int numOfDoors;
    string color;

    void accelerate()
    { \\ logic for acceleration }

        void brake()
    { \\ logic for brakes }


};
```

# How to use the car?

```
int main()
{
    Car mycar;
    car.accelerate();
}
```

# Class vs Struct

- Members of a class are **private** by default and members of struct are **public** by default
- Classes can be inherited whereas structures cannot
- Struct are value-type whereas classes are reference-type
- A structure can't be abstract, a class can be

# Problem

```
int main()
{
    Car mycar;
    car.accelerate();
}
```

**The function accelerate() cannot be accessed!**