PEMROGRAMAN LANJUT

13 | Interface

Team Teaching Pemrograman Lanjut

Fakultas Ilmu Komputer UB

Abstract Base Class

- Suatu class yang hanya dapat digunakan sebagai base class yang memiliki fungsi virtual murni (pure virtual function)

```
// abstract class CPolygon
class Polygon {
  protected:
    int width, height;
  public:
    void set_values (int a, int b)
      { width=a; height=b; }
    virtual int area () =0;
};
```

- Perhatikan kode di atas, bahwa `area()` tidak memiliki definisi (implementasi) yang ditandai dengan "=0" (pure virtual function)

Class yang memiliki setidaknya 1 pure virtual function disebut sebagai Abstract Base Class

Abstract Base Class

- Tidak dapat diinstansiasi secara langsung
- Memiliki keuntungan dalam pembuatan polimorfisme

Interface vs Abstract Class

- Interface tidak memiliki data member dan implementasi method (seluruhnya PVF)
- Derived class dari interface harus mengimplementasikan seluruh method pada interface
- Abstract dapat berisi data member dan implementasi method (setidaknya 1 PVF)
- Derived class dari abstract tidak harus mengimplementasikan method (kecuali PVF)

Interface vs Abstract Class

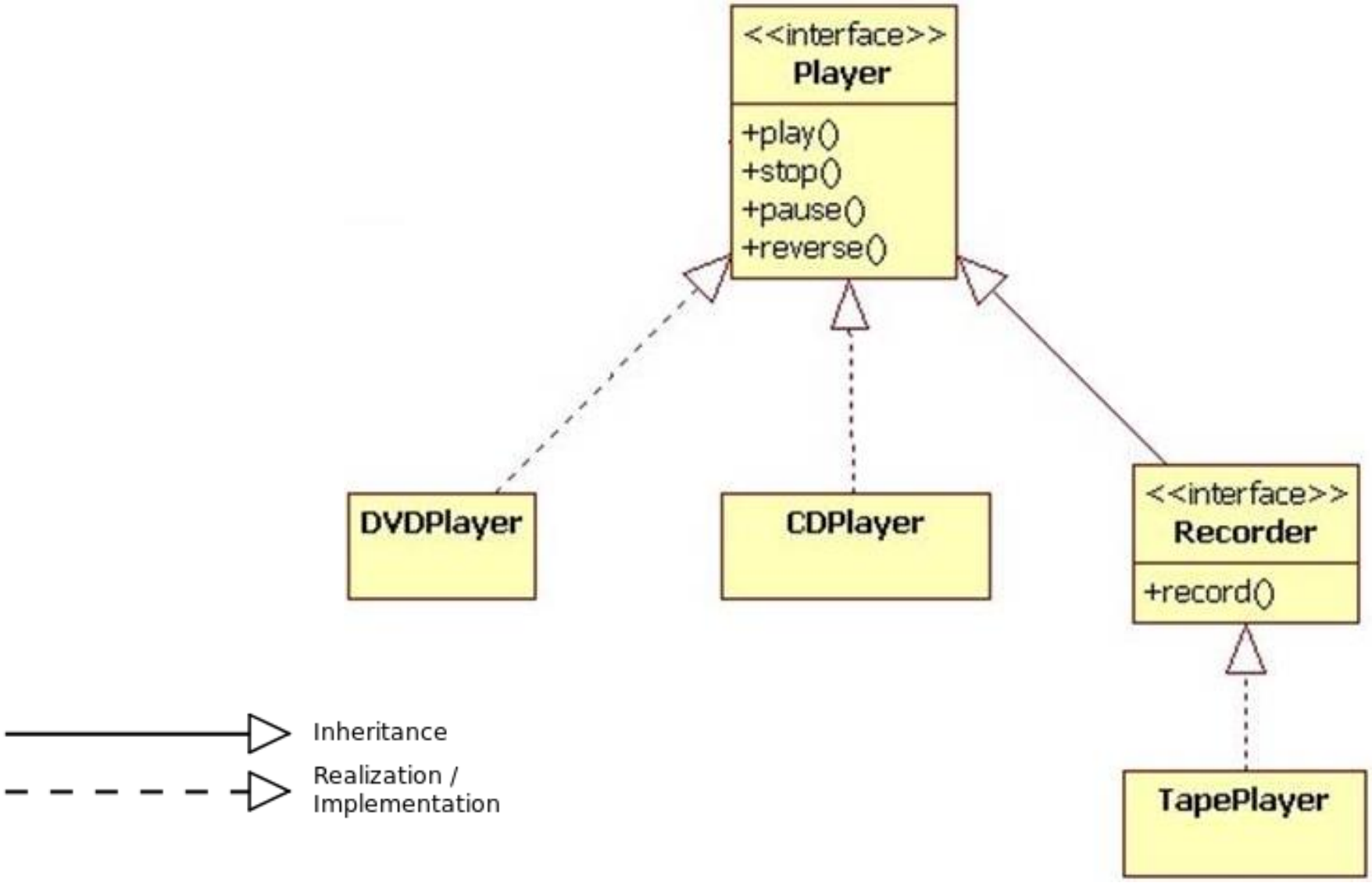

```
class MyInterface
{
public:
  // Empty virtual destructor for proper cleanup
  virtual ~MyInterface() {}

  virtual void Method1() = 0;
  virtual void Method2() = 0;
};
```

```
class MyAbstractClass
{
public:
  virtual ~MyAbstractClass();

  virtual void Method1();
  virtual void Method2();
  void Method3();

  virtual void Method4() = 0; // make MyAbstractClass not instantiable
};
```

PERHATIAN!!!

- Secara teknik, **C++ TIDAK memiliki istilah INTERFACE**, hanya sebuah konsep

Class Diagram

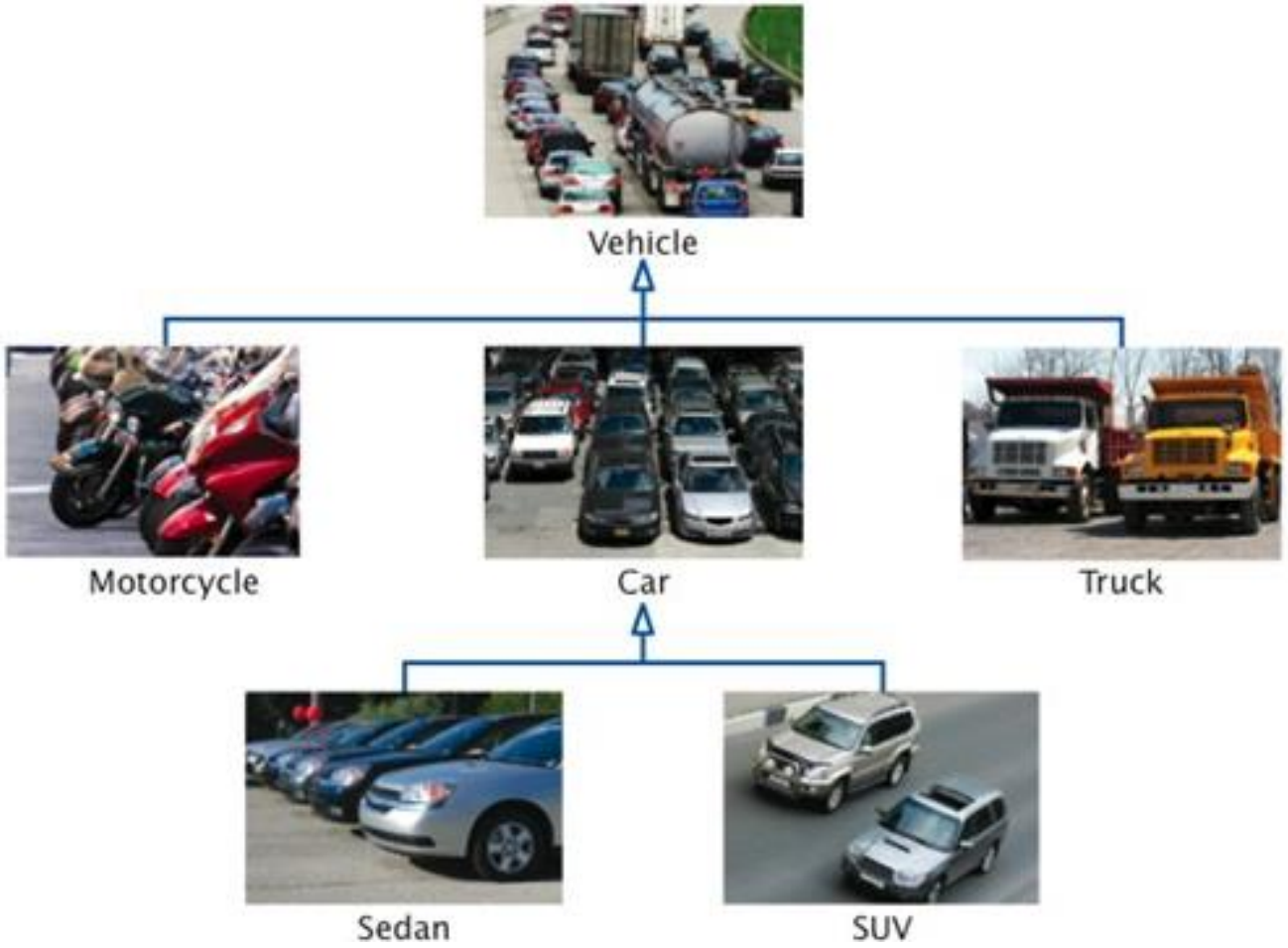


Code


```
class Player {
public:
   virtual void play() = 0;
   virtual void stop() = 0;
   virtual void pause() = 0;
   virtual void reverse() = 0;
};

class Recorder: public Player {
public:
   virtual void record() = 0;
};
```

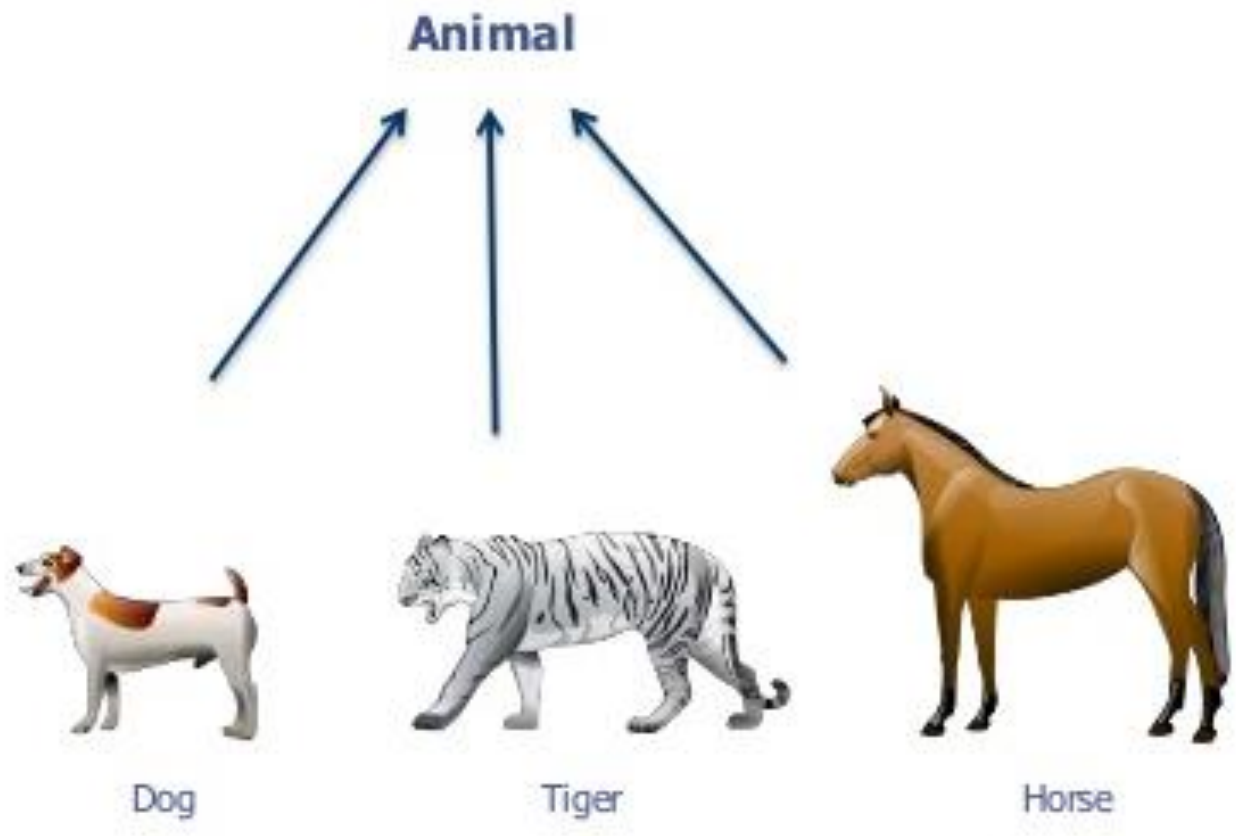
```
class TapePlayer: public Recorder {
public:
   void play();
   void stop();
   void pause();
   void reverse();
   void record();
};

void TapePlayer::play() { ... }
void TapePlayer::stop() { ... }
void TapePlayer::pause() { ... }
void TapePlayer::reverse() { ... }
void TapePlayer::record() { ... }
```

Kapan Menggunakan Interface & Abstract?

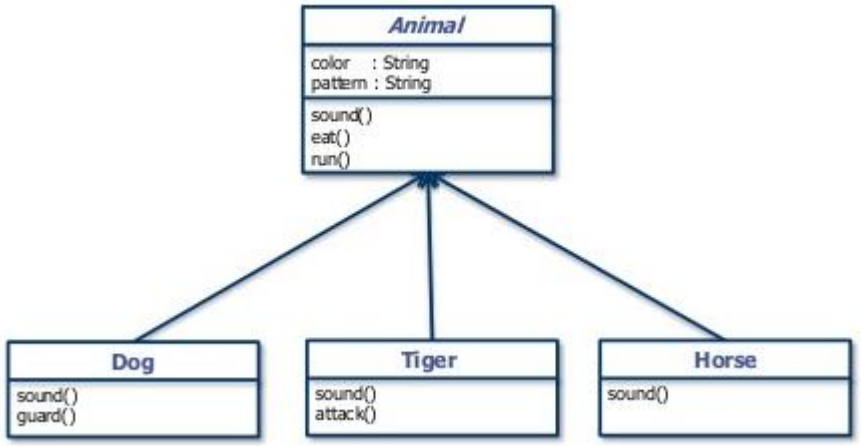


ABSTRACT

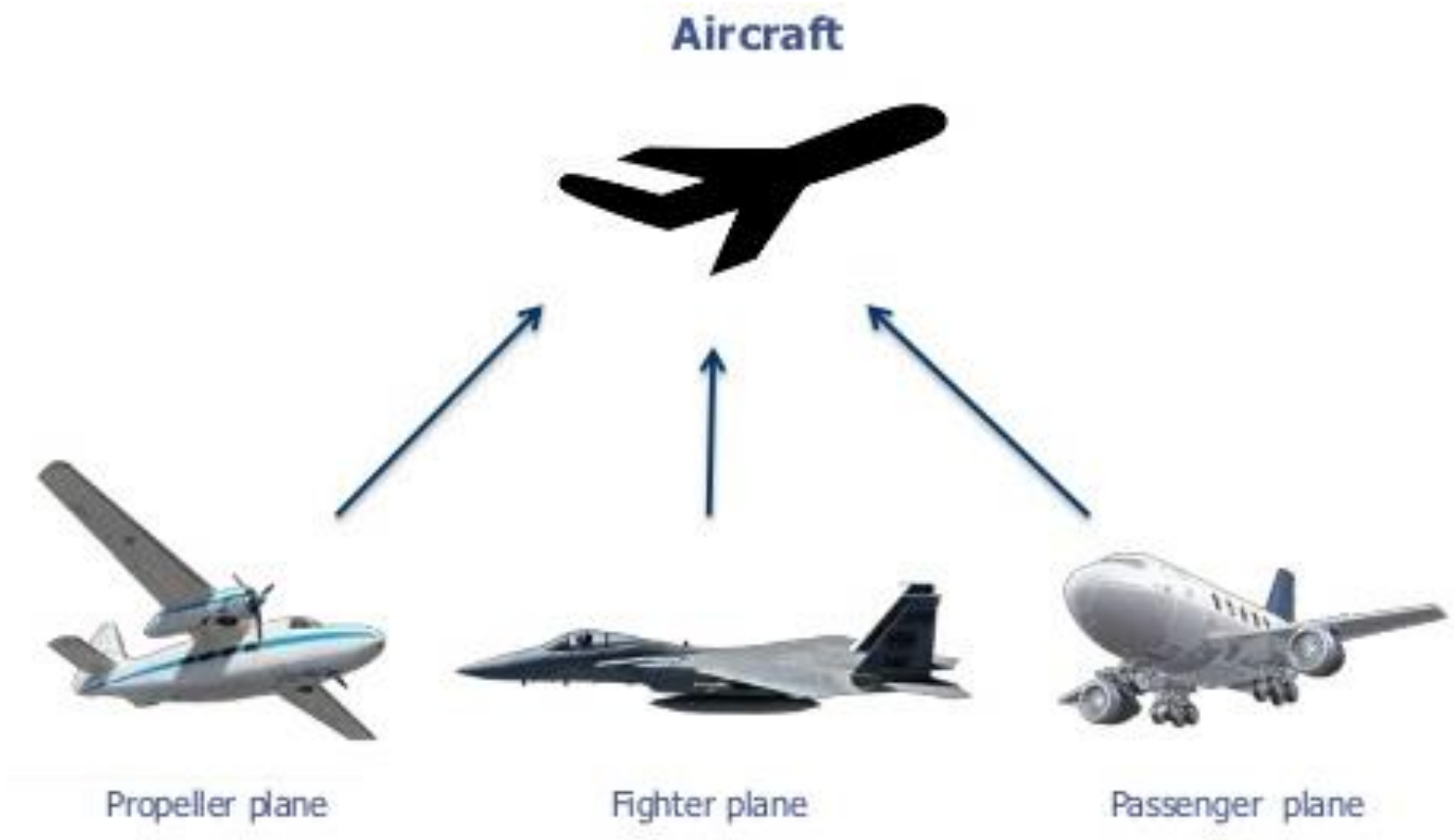
Kapan Menggunakan Interface & Abstract?



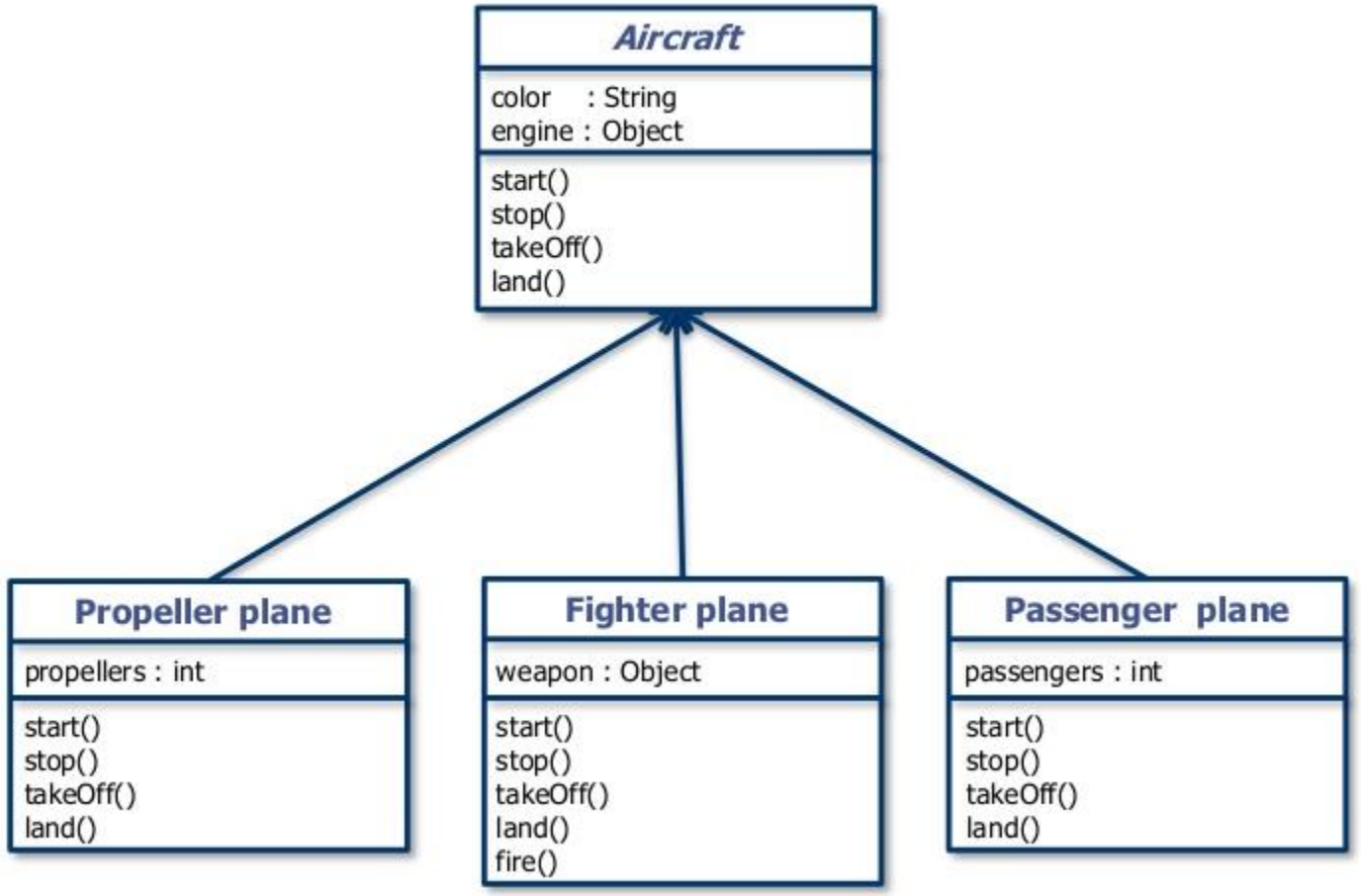
ABSTRACT



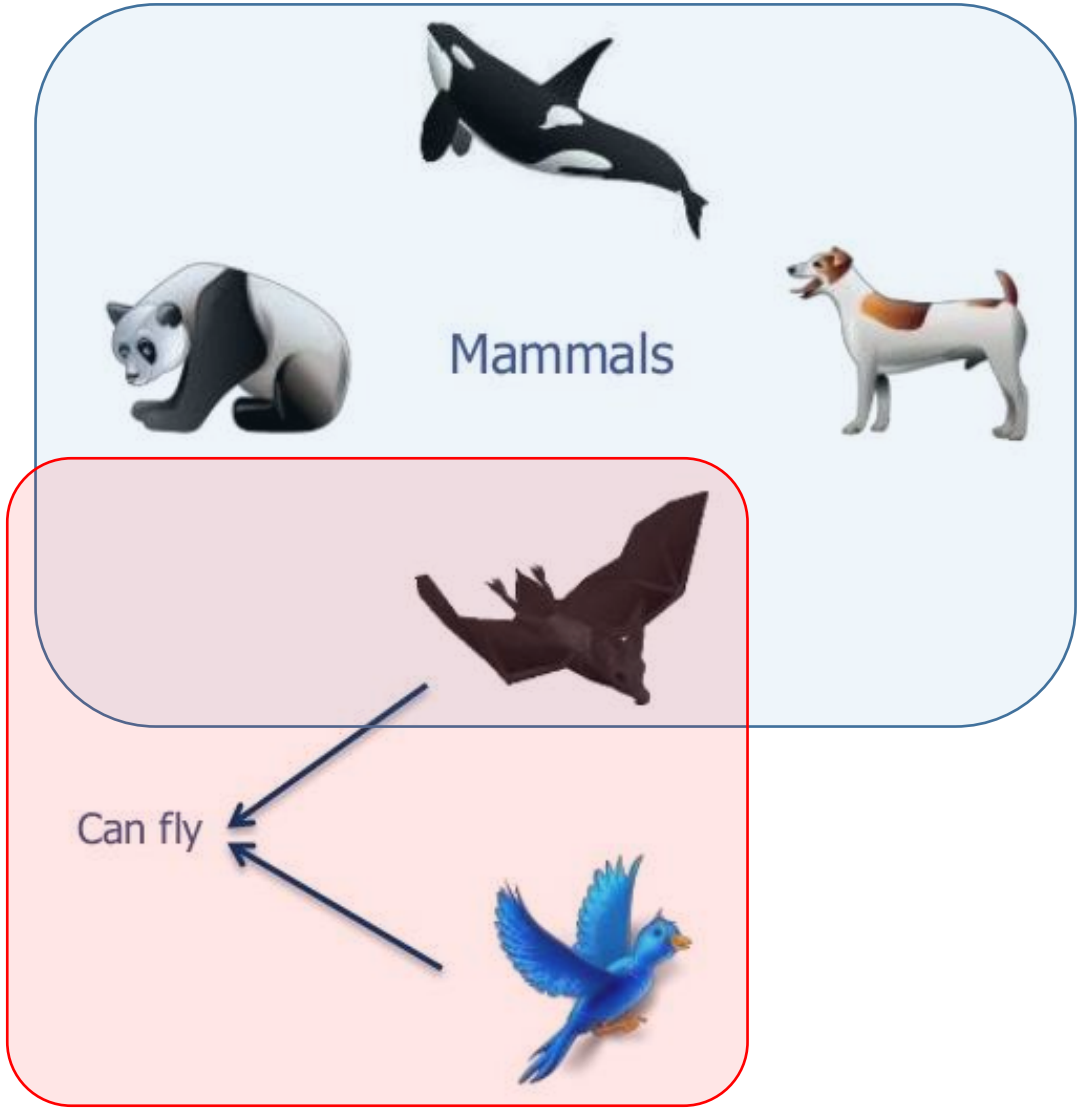
Kapan Menggunakan Interface & Abstract?



ABSTRACT


Aircraft
color : String
engine : Object
start()
stop()
takeOff()
land()
Propeller plane
propellers : int
start()
stop()
takeOff()
land()
Fighter plane
weapon : Object
start()
stop()
takeOff()
land()
fire()
Passenger plane
passengers : int
start()
stop()
takeOff()
land()

Kapan Menggunakan Interface & Abstract?

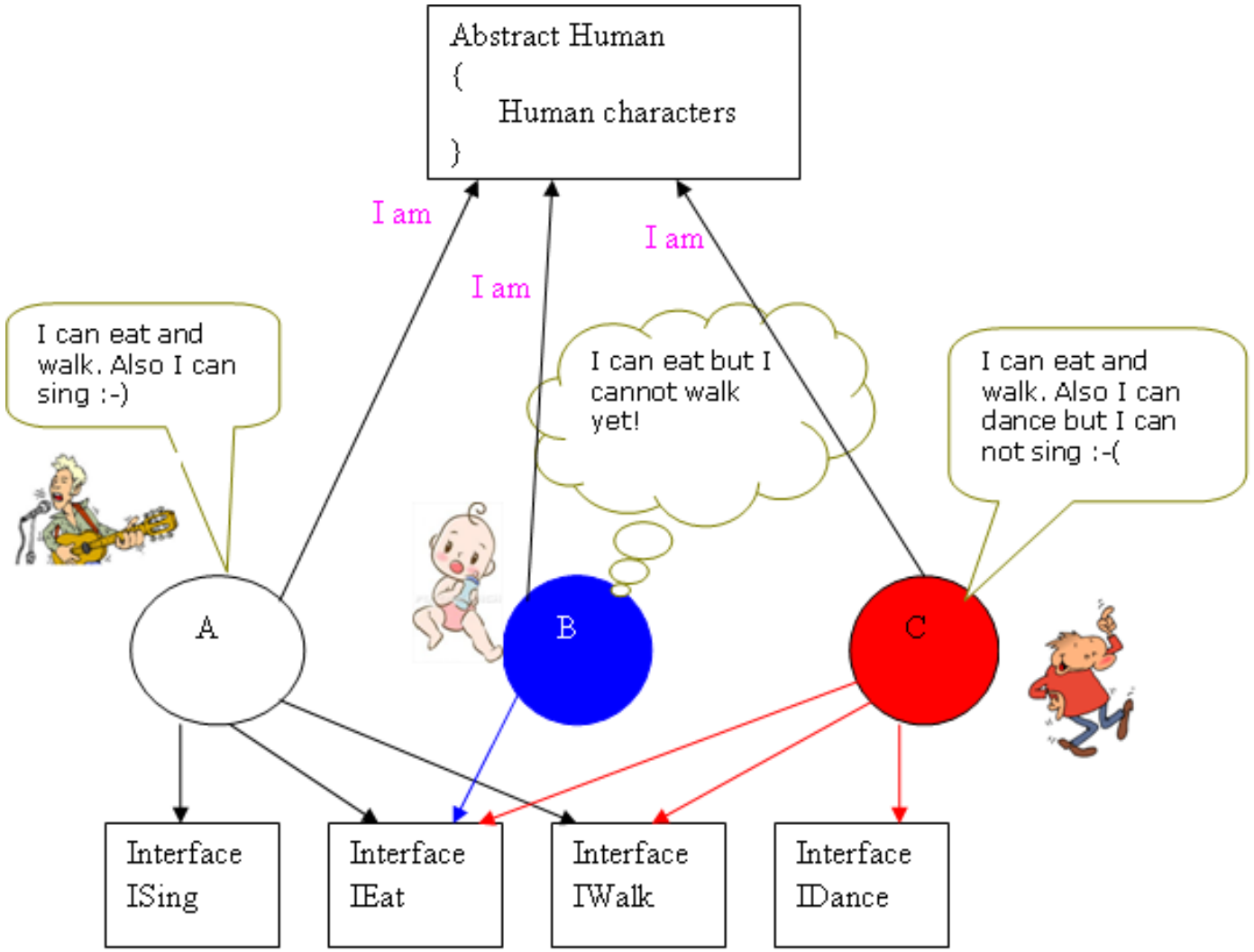


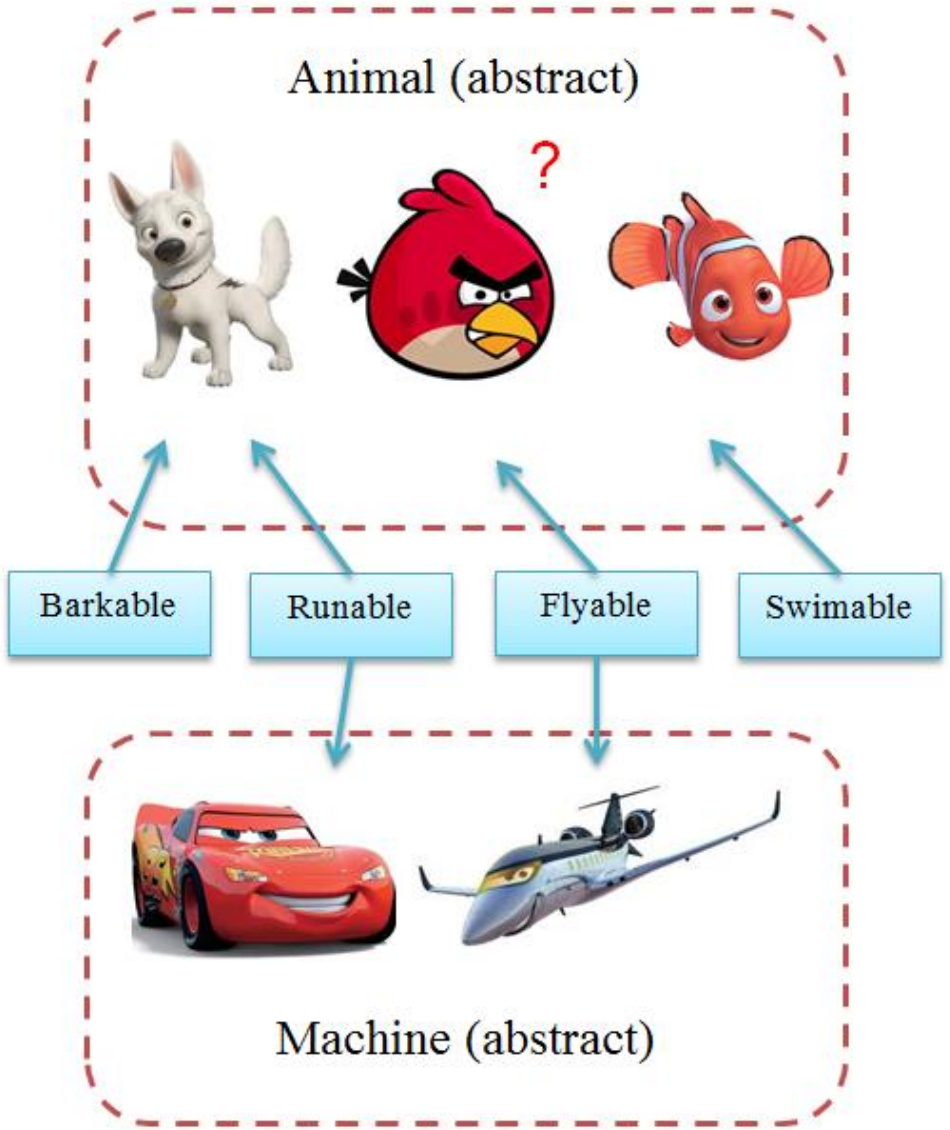
ABSTRACT

Abstract memiliki relasi "**is-a**"

INTERFACE

Interface memiliki relasi "**can-do**"


Abstract Human
{
Human characters
}
I am
I am
I am
I can eat and walk. Also I can sing :-)
I can eat but I cannot walk yet!
I can eat and walk. Also I can dance but I can not sing :-(
A
B
C
Interface ISing
Interface IEat
Interface IWalk
Interface IDance


Animal (abstract)
?
Barkable
Runable
Flyable
Swimable
Machine (abstract)

Mengapa Interface?

- **Kelemahan** multiple inheritance adalah suatu class bisa **mewarisi method** lebih dari satu class dimana method ini **tidak diharapkan**. Dengan Interface maka hal ini bisa dihindari
- Objek yang tidak berelasi memiliki method yang sama
- Tidak membutuhkan implementasi dari base class