**#13 OOP DASAR pada PHP Abstract Class**

Konsep class abstract ini salah satu konsep yg agak sulit untuk dipahami ya, kalo nanti agak sulit untuk dipahami bisa ulang aja video nya. Untuk video ini kita akan membahas mengenai Class Abstract nya saja dan di video selanjut nya baru praktek nya.

* Sebuah kelas yg nantinya tidak dapat kalian instansiasi. Jadi kita tidak bisa bikin object dari class abstract itu. Nanti kita instasiasi nya dengan class turunan nya.
* Kelas *‘abstrak’* artinya tidak dapat di instasiasi. Yg nanti bisa jadi class sesungguhnya adalah class-class turunan nya
* Class ini akan mendefinisikan interface untuk kelas turunan nya.
* Class ini nantinya akan berperan sebagai *‘kerangka dasar’* untuk kelas turunannya. ( jadi nnti class turunan nya akan bekerja sesuai kerangka atau interface yg sudah kita buat di class abstract.
* Didalam class ini memiliki minimal 1 **method abstrak.** nanti method ini akan kita anggap sebagai interface atau kerangka dari method yg akan kita buat di class-class turunan nya.
* Digunakan dalam ‘pewarisan’ / ‘inheritance untuk **‘memaksakan’** implementasi method abstrak yang sama untuk semua kelas turunannya. Jadi didalam kelas abstrak nanti kita punya method abstrak yg hanya interface nya aja, yg nama nya aja tidak ada isi nya. Nah isi nya nanti kita tuliskan di clas-class turunan nya dengan menggunakan yg nama nya sama.

Kalian bisa liat bahwa class abstrak ini sangat erat kaitan nya dengan inheritance atau pewarisan.

**Contoh kasus :**

class Buah {

    private $warna;

    public function makan() {

*// kunyah*

*// nyam..nyam..nyam*

    }

    public function setWarna($warna) {

**$this**->warna = $warna;

    }

}

Jadi saya punya 2 method satu nya setWarna() untuk mengubah property $warna dan ada method makan() yg ngasih tau cara nya saya makan buah ini, tapi disitu ceritanya saya ga definisikan di method makan() nya ya.

Selanjut nya saya punya sebuah class sebagai class turunan dari class buah itu :

class Apel extends Buah {

    public function makan() {

*// kunyah*

*// sampai bagian tengah*

    }

}

Jadi class apel bisa akses semua property dan method yg ada di buah kecuali visibility nya Private ya. Di class apel saya buat method nama nya sama yaitu method makan(), dan saya juga punya class turunan nya lagi :

class Jeruk extends Buah {

    public function makan() {

*// kupas*

*// kunyah*

    }

}

Misal kalian saya kasih buah apel terus saya suruh kalian makan apel itu :

$apel = new Apel();

$apel->makan();

Terus sekarang saya tanya rasa nya apa pasti kalian akan jawab rasa apel, jadi kalo kaya gitu gaada masalah, tapi sekarang coba saya kasih kalian buah gini :

$buah = new Buah();

$apel->makan();

Sekarang pertanyaan nya BUAH nya rasa apa ? kalia pasti bingung jawab nya kan harus nya buah apa dulu gitu apakah apel atau jeruk. Nah jadi seperti nya Buah itu buka sebuah class yg akan kita instasiasi.

Nah kalo ada kasus kaya gitu itu adalah hal yang tepat untuk menggunakan abstract, nah jadi class buah akan kita buat menjadi class abstract berarti nanti nya tidak bisa kita instasiasi.

Membuat Kelas Abstrak

Cara membuat kelas bastrak kita tinggal menambahkan keyword **abstract** di awal class nya gini

abstract class Buah {

}

Lalu isi nya kita tulis gini :

abstract class Buah {

    private $warna;

    abstract public function makan();

    public function setWarna($warna) {

**$this**->warna = $warna;

    }

}

Nah lihat kan minimal ada method abstrak satu nah disitu ada satu method yaitu makan() yg kita tulis hanya interface nya aja isi nya nanti di class turunan nya seperti class jeruk dan apel.

Nah ini membuat class turunan nya wajib punya implementasi dari method tadi, kalo ada misal buat class durian yg nyambung ke class buah maka didalam nya wajib ada implementasi dari method makan().

Selain itu pun ada konsep yg lain **Abstract Class (2)** :

* Semua class turunan, harus mengimplementasikan method abstrak yg ada di kelas abstraknya.
* Boleh memiliki property / method biasa
* Boleh memiliki property atau static method

Contoh Kelas Abstrak

* Class Mobil Extends Kendaraan
* Class Laptop Extends Komputer
* Class Email Extends Komunikasi
* Dll.

Kenapa menggunakan kelas abstrak ?

* Merepresentasikan ide atau konsep dasar yg nanti nya akan di implementasikan di kelas turunan nya.
* Dan yg harus kalian pahami penggunaan class abstrak ini bukan masalah bener dan salah nya tapi tentang keputusan desain nya, perancangan dari class, kalo ingin menggunakan OOP kenapa tidak pake saja kelass abstrak ini.
* “Composition over Inheritance” jadi sebaik nya kita melakukan composisi dibandingkan kita melakukan inharetance begitu saja. sebetul nya tanpa kelas abstrak pun bisa hanya menggunakan inheritance saja.
* Dengan melakukan komposisi ini merupakan salah satu carak menerapkan Polimorphism ini yg lebih kompleks lagi, tapi untuk nyampe ke situ kalian harus paham dulu mengenai abstrak dan interface.
* Melakukan abstraksi itu kalian bisa lakukan sentralisasi logic
* Dan juga dengan melakukan kelas abstrak mempermudah pengerjaan tim

Dengan menbuat kelas abstrak kalian sudah membuat base class yaitu class utama nanti didalam class itu kalian tulis beberapa method abstrak sehingga nanti ada rekan tim kalian akan membuat sub class atau class turunan nya mereka udah tau didalam class turunan nya wajib ada sebuah method sebagai implementasi dari method abstrak nya.