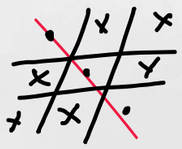
OOP in PHP

Class → object

OBJECTS CAN DERIVATE FROM CLASSES THROUGH **INSTANTIATION**

OOP MODELS REALITY BY MEANS OF "**CLASSES**" DEFINING:

- **PROPERTIES** (=CHARACHTERISTICS)
- **METHODS** (=BEHAVIOURS)



object:

- IS MADE OF → PROPERTIES (DATA), METHODS (FUNCTIONS)
- IS A **CUSTOMIZABLE COPY** OF ITS PARENT CLASS
- CAN BE INITIALIZED (CUSTOMIZED) BY USING **__CONSTRUCT**
- HAS A **UNIQUE INSTANCE** IDENTIFIED BY PSEUDO-VARIABLE **$THIS**

CREDITS

Content
PASQUALE ALFANO
linkedin.com/in/pasqualealfano/
@ibisreclibis
github.com/halfpass25/

Graphic
LORENZO ALFANO
linkedin.com/in/lorenzoalfano/
behance.net/LorenzoAlfano

The four pillars of OOP

Encapsulation

HIDES CLASSES INTERNAL DETAILS BUT **EXPOSE** PROPERTIES (CHARACTERISTICS A.K.A. DATA) AND METHODS (=FUNCTIONS) VIA A PUBLIC INTERFACE, PREVENTING UNINTENTIONAL MODIFICATION OF PROPERTIES (DATA) VALUES.

VISIBILITY (FOR PROPERTIES AND METHODS)

- **Public** — AVAILABLE IN THE **GLOBAL SCOPE** FOR R/W
- **Protected** — AVAILABLE **ONLY** TO **INHERITING** CLASSES
- **Private** — AVAILABLE **ONLY** TO THE **PARENT** CLASS

Inheritance

A CLASS CAN "INHERIT" PROPERTIES (DATA) AND METHODS (FUNCTIONS) FROM OTHER CLASSES. THIS ALLOWS FOR REUSABLE AND WELL ORGANIZED CODE. A CLASS, BESIDES INHERITING , CAN DEFINE ITS OWN PROPERTIES AND METHODS.

!

BUT PHP OFFER **SINGLE INHERITANCE**:
HOW TO OVERCOME SUCH A LIMITATION?

YOU CAN USE THE FOLLOWING TECHNIQUES **IN ISOLATION OR COMBINED AT ONCE** TO OVERCOME SINGLE INHERITANCE LIMITED CAPABILITIES

Vertical inheritance

(KEYWORD: **EXTENDS**) IS THE BASE INHERITANCE OFFERED BY PHP. TO TELL A CHILD CLASS IT WILL INHERIT FROM A PARENT ONE, YOU USE THE "EXTENDS" KEYWORD.

Horizontal inheritance

A.K.A. "TRAITS" (KEYWORD: **USE**) BY MEANS OF TRAITS A CHILD CLASS CAN REUSE CODE FROM MULTIPLE CLASSES WITHOUT REQUIRING THE LATTER BEING CHILDREN OF A PARENT CLASS. TO PUT TRAITS AT WORK:

1. DEFINE A TRAIT (TIPYCALLY, IS A COLLECTION OF METHODS)
2. TO INCLUDE IT INTO A CLASS, USE THE "USE" KEYWORD
3. METHODS EXPOSED BY TRAITS BECOME AVAILABLE TO THE CLASS AS IF THEY WERE DEFINED WITHIN THE CLASS ITSELF.

Interface

AN INTERFACE –IN PHP–, SPECIFIES A SET OF METHODS THAT MUST BE IMPLEMENTED BY ANY CLASS THAT ADHERES TO IT, HENCE, IS A WAY TO DEFINE A "CONTRACT". WITHIN AN "INTERFACE" YOU DECLARE WHAT METHODS A CLASS SHOULD EXPOSE AND THEN YOU PROVIDE SPECIFIC IMPLEMENTATION OF EACH SPECIFIC METHOD WITHIN ADHERING CLASS. IF IN THE INTERFACE YOU DECLARED A METHOD CALLED "CALCULATEAREA", YOU CAN HAVE CLASSES CALCULATING A CIRCLE OR A SQUARE AREA AND EACH CLASS WILL IMPLEMENT THE "CALCULATEAREA" METHOD, ACCORDING TO THE GEOMETRIC SHAPE IT DOES REFER. THE NAME OF THE METHOD STAY THE SAME (CALCULATEAREA) BUT ITS IMPLEMENTATION CHANGES BASED ON THE SPECIFIC NEEDS.

Polymorphism

ENABLES OBJECT TO CHANGE THEIR FORM SIMPLY EXTENDING AND/OR OVERRIDING ALREADY EXISTING METHODS (E.G.: INDIAN AND BRITISH CAN EXTEND SPEAKER BUT WILL HAVE THEIR OWN SPOKEN_LANGUAGE).

Abstraction

ABSTRACTION IS MEANT TO REDUCE COMPLEXITY BY SHOWING ONLY WHAT IS NEEDED IN A GIVEN CONTEXT. AS A STATEMENT OF EXAMPLE, A USER DOESN'T NEED TO KNOW HOW AN ATM MACHINE INTERNALLY WORKS: TO HIM/HER, INTERACTING WITH THE ATM IS ENOUGH.

SOURCE: WWW.GITHUB.COM/HALFPASS25