UNAERP

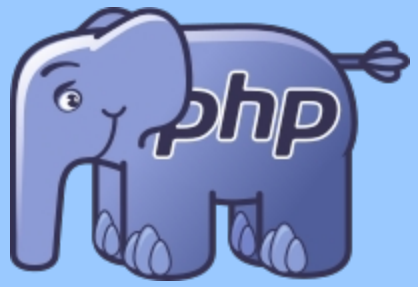
Universidade de Ribeirão Preto

Campus Ribeirão Preto - Campus Guarujá

UNAERP
Universidade de Ribeirão Preto
Campus Ribeirão Preto - Campus Guanajá

Programação para Web II

POO - SOLID



Prof. Eliézer Zarpelão

Schedule

- Encapsulation
- Inheritance
- Polymorphism
- Abstract Class
- Interfaces
- Composition
- Agregattion
- Class Members
- SOLID

Encapsulation

The Cat class has mood, hungry and engergy private fields and a meow private method

Cat

- mood - field
- hungry - field
- energy - field
- meow() - method

Feed, Play and Sleep are public methods. Other classes can call them, but they can't directly modify the private fields.

Sleep

- energy ++
- hungry ++

Play

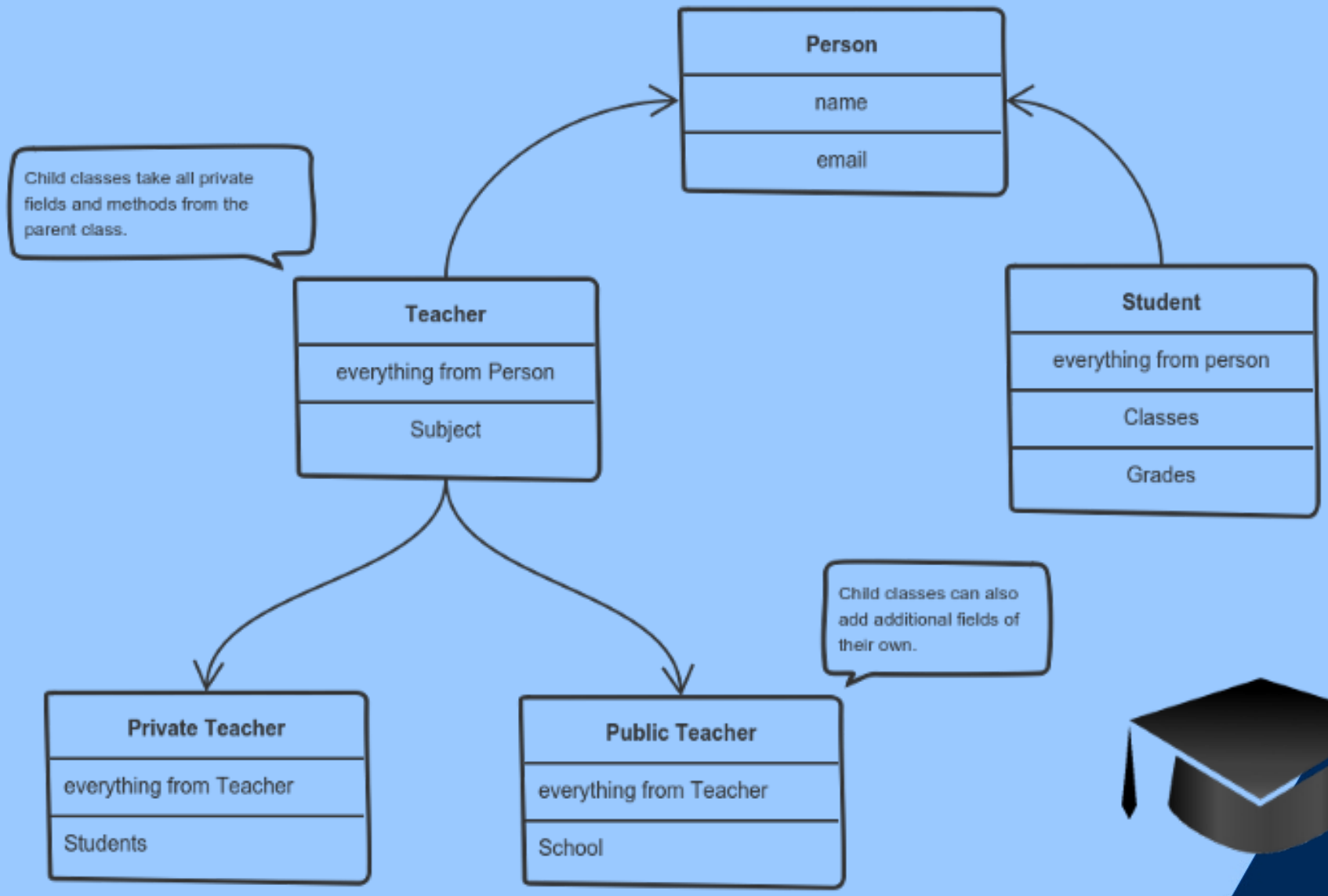
- mood ++
- energy --
- meow!

Feed

- hungry --
- mood ++
- meow!

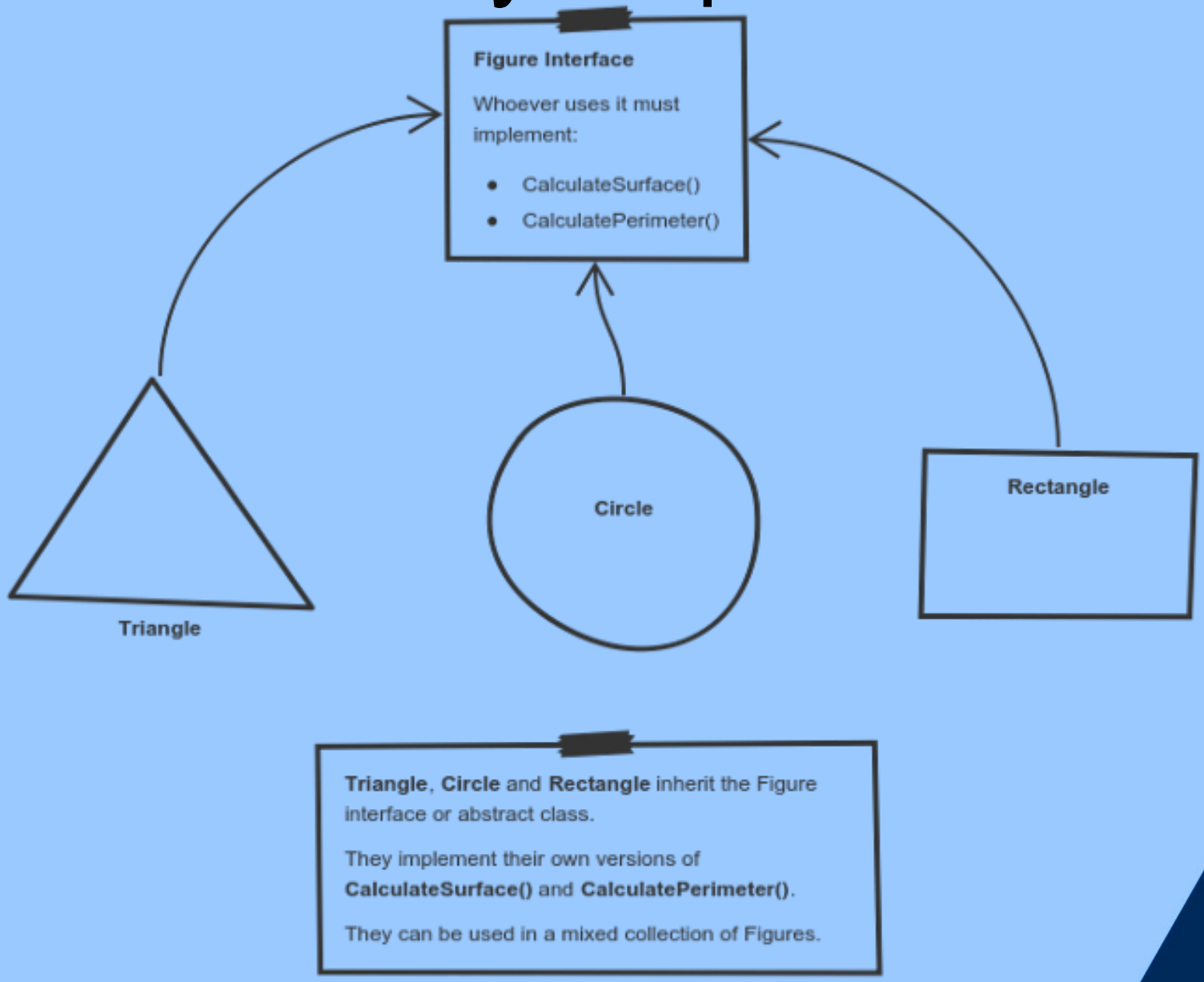
https://www.freecodecamp.org/news/object-oriented-programming-concepts-21bb035f7260

Inheritance



https://www.freecodecamp.org/news/object-oriented-programming-concepts-21bb035f7260

Polymorphism



https://www.freecodecamp.org/news/object-oriented-programming-concepts-21bb035f7260

Abstract class

- classes that may contain abstract method(s)
  - abstract method is a method that is declared, but contains no implementation
- can’t be instantiated
- require subclasses to provide implementations for the abstract methods

Interfaces

- Enforce certain properties on an object
- Interface is very similar to an abstract class, but it has **no properties** and cannot define how methods are to be implemented.
  - Instead, it is simply a list of methods that **must** be implemented
- Interface is a group of related methods with empty bodies

Composition

- Object creates another object
- ownership relationship

Aggregation

- Object is composed of multiple objects
- “contains” relationship

Class Members

- Constants
- Statics attributes
- Statics methods

SOLID

- S — Single Responsiblity Principle
- O — Open-Closed Principle
- L — Liskov Substitution Principle
- I — Interface Segregation Principle
- D — Dependency Inversion Principle

SINGLE RESPONSIBILITY PRINCIPLE

Just Because You Can, Doesn't Mean You Should

Single Responsibility Principle

- A class should have one and only one reason to change, meaning that a class should have only one job.

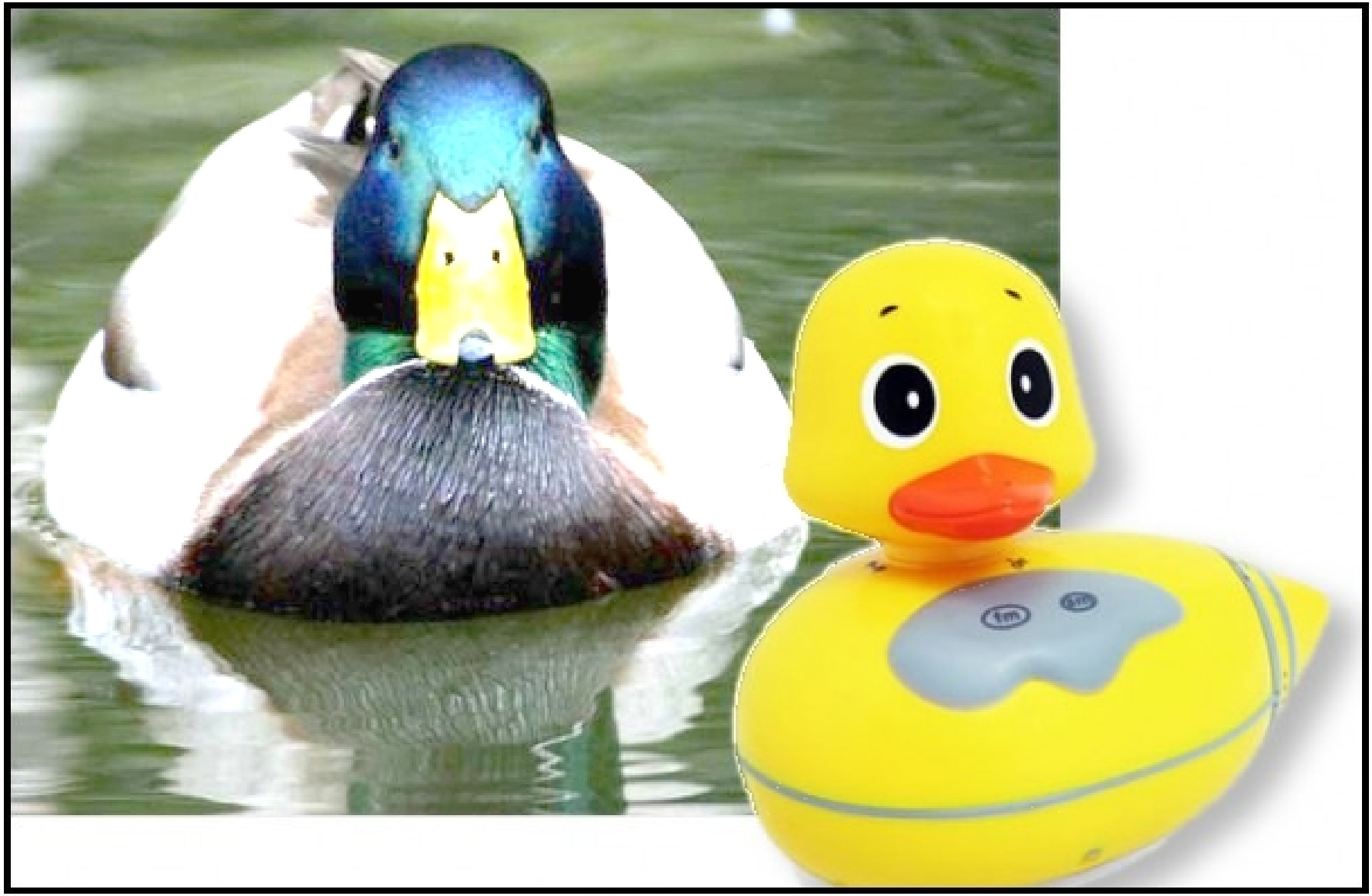
http://www.lostechies.com/blogs/derickbailey/archive/2009/02/11/solid-development-principles-in-motivational-pictures.aspx

OPEN CLOSED PRINCIPLE

Open Chest Surgery Is Not Needed When Putting On A Coat

Open-Closed Principle

- Objects or entities should be open for extension, but closed for modification.

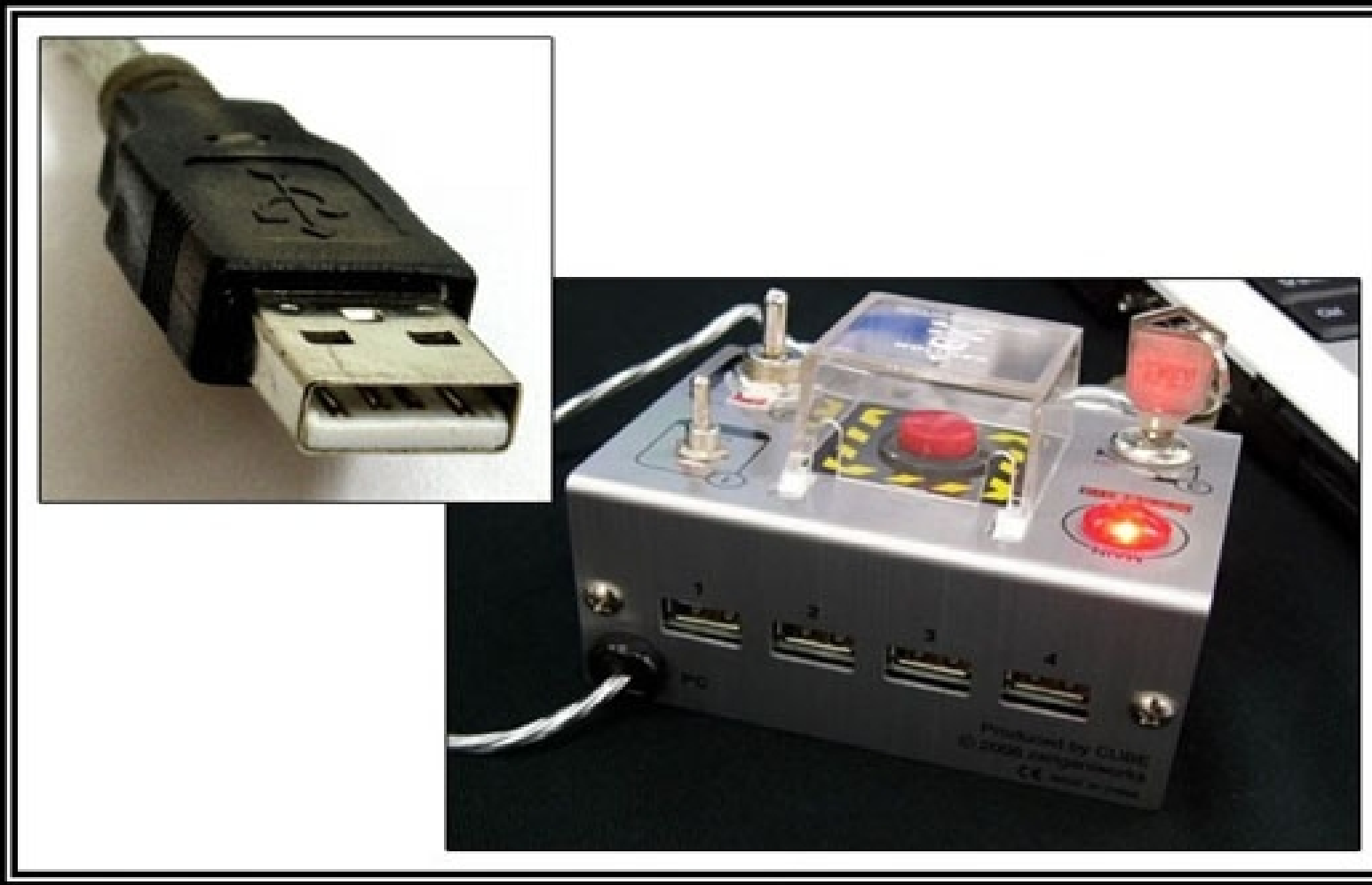


LISKOV SUBSTITUTION PRINCIPLE

If It Looks Like A Duck, Quacks Like A Duck, But Needs Batteries - You Probably Have The Wrong Abstraction

Liskov Substitution Principle

- Let q(x) be a property provable about objects of x of type T. Then q(y) should be provable for objects y of type S where S is a subtype of T.



INTERFACE SEGREGATION PRINCIPLE

You Want Me To Plug This In, Where?

Interface Segregation Principle

- A client should never be forced to implement an interface that it doesn't use or clients shouldn't be forced to depend on methods they do not use.

DEPENDENCY INVERSION PRINCIPLE

Would You Solder A Lamp Directly To The Electrical Wiring In A Wall?

Dependency Inversion Principle

- Entities must depend on abstractions not on concretions. It states that the high level module must not depend on the low level module, but they should depend on abstractions.

Exercise

- Object Calisthenics
- Kales → Good / Simple
- Write/explain the (9) rules and **make** examples with **original** PHP code
- Send by e-mail to ezarpelao@unaerp.br
- Deadline: 23/09/2019
- Extra point – AV1
- Individual

Doubts

[Maior dúvida] – send by e-mail to ezarpelao@unaerp.br subject "Maior dúvida – 06/09/2019 - "+RA

Deadline: 09/09/2019