고급진 객체지향 개발론

OOP 소개

daumkakao
최윤상

OOP

Object Oriented Programming

What is **Programming**?

Software Developers

what my friends thinks I do

what my mum thinks I do

what users think I should do

what testers thinks I do

what I think I do

what I really do

Programmer

What my friends think I do

What my mom thinks I do

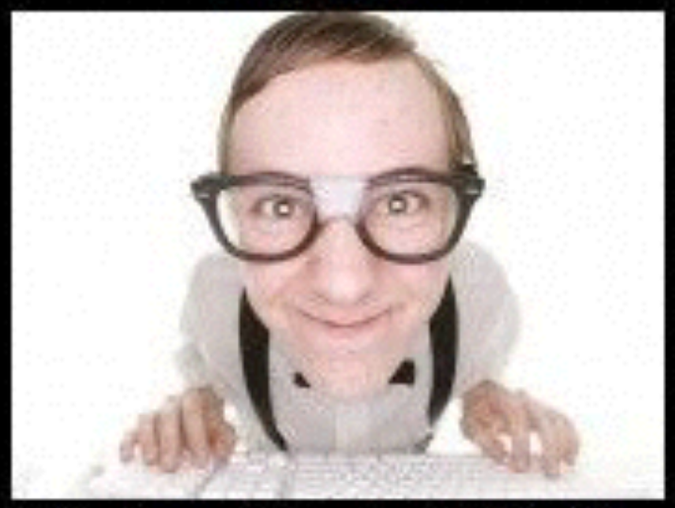
What society thinks I do

What my boss thinks I do

What I think I do

What I actually do

Software Development



What my friends think I do

What my family thinks I do

What users think I do

What users think I SHOULD do

What I think I do

What I actually do

현실 속의

문제를 해결하는 과정

컴퓨터를 이용해서

현실 속의

주어진 문제를
어떻게 **모델링**할 것인가?

OOP 역시

문제를 **모델링**하는

방법 중 하나

OOP의 주요 개념

문제를 모델링하는 방법

Object

속성과 행위, 그리고 아이덴티티

Class

객체생성을 위한 청사진

Encapsulation

노출할 것과 감출것을 결정

Inheritance

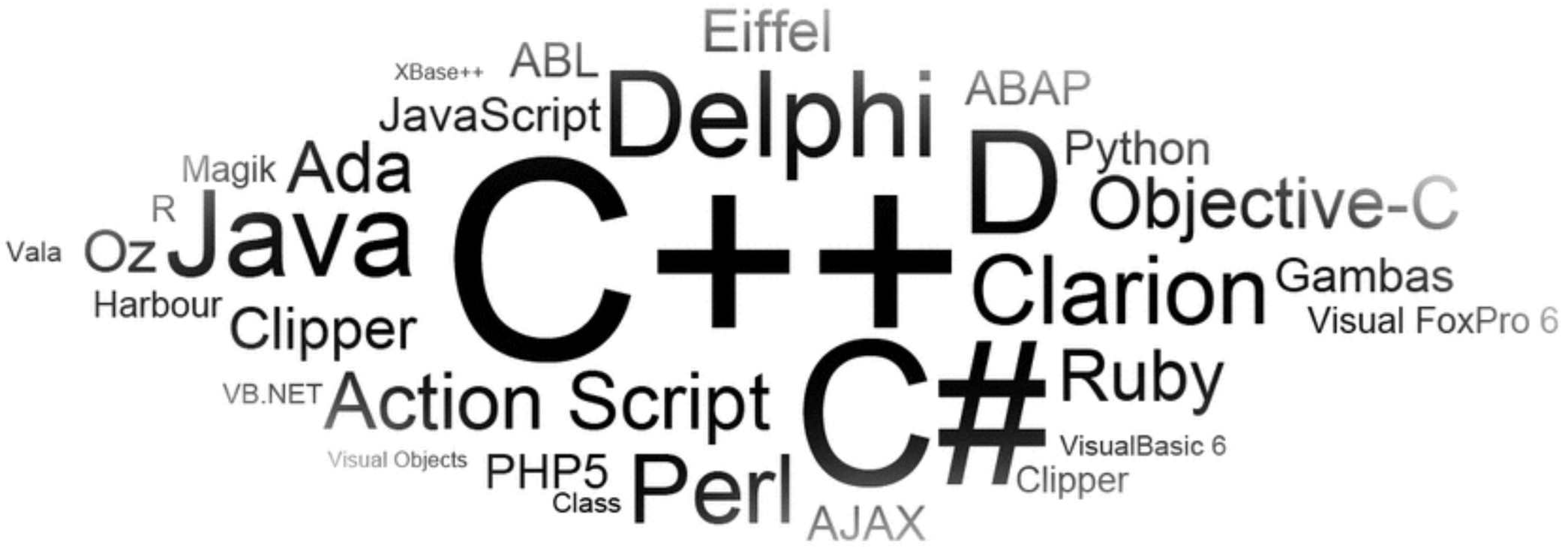
클래스의 계층구조

Polymorphism

흐름 제어

새끼를 낳으면 “포유류”
척추가 있으면 “척추동물”

Object, Class, Encapsulation,
Inheritance, Polymorphism
이런걸 지원하면 **OOP 언어**라고 함


Eiffel
XBase++
ABL
JavaScript
Delphi
ABAP
Magik
Ada
Python
R
Java
C++
D
Objective-C
Vala
Oz
Clarion
Gambas
Harbour
Clipper
Visual FoxPro 6
VB.NET
Action Script
C#
Ruby
Visual Objects
PHP5
Class
Perl
VisualBasic 6
Clipper
AJAX

OOP의 약속들?!

real world를 모델링할 수 있다

non-OOP로는 실세계를 모델링 못하나?

코드재사용성을 높여
품질을 높이고 비용을 줄일 수 있다

non-OOP로 개발하면 코드재사용성이 떨어지는가?

모듈별로 작성되고 유지보수할 수 있다

non-OOP는 모듈화가 안되는가?

그럼
Java로 프로그래밍하면
객체지향프로그래밍일까?

Java로 개발하면
C를 사용할 때보다
모듈화, 코드재사용성등이
더 좋아질까?

OOP Design Principles

https://lostechies.com/derickbailey/2009/02/11/solid-development-principles-in-motivational-pictures/

SOLID 원칙

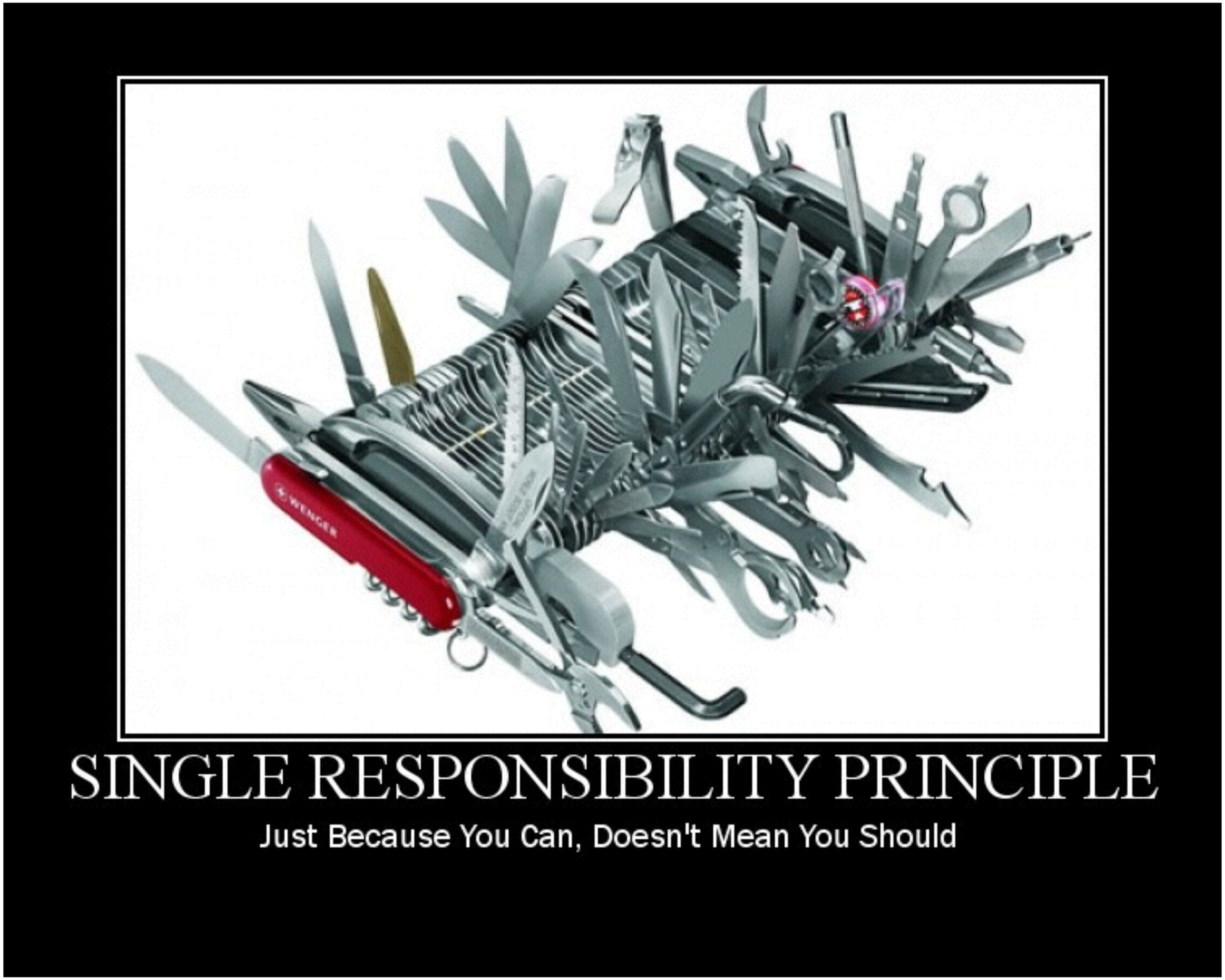
Single Responsibility Principle (SRP)

Open Closed Principle (OCP)

Liskov Substitution Principle (LSP)

Interface Segregation Principle (ISP)

Dependency Inversion Principle (DIP)

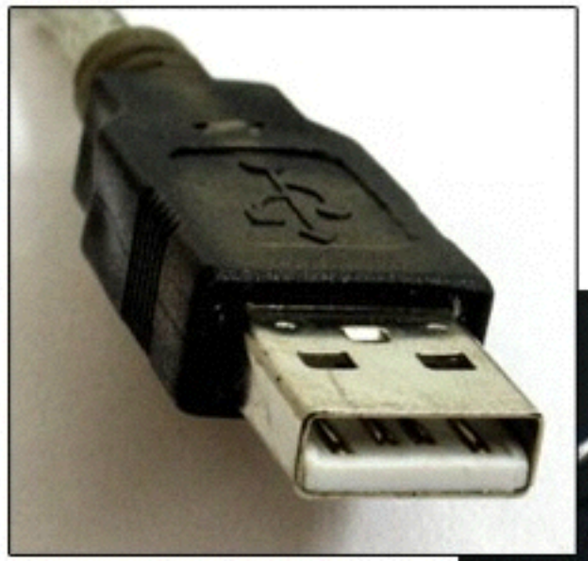

SINGLE RESPONSIBILITY PRINCIPLE
Just Because You Can, Doesn't Mean You Should

OPEN CLOSED PRINCIPLE

Open Chest Surgery Is Not Needed When Putting On A Coat

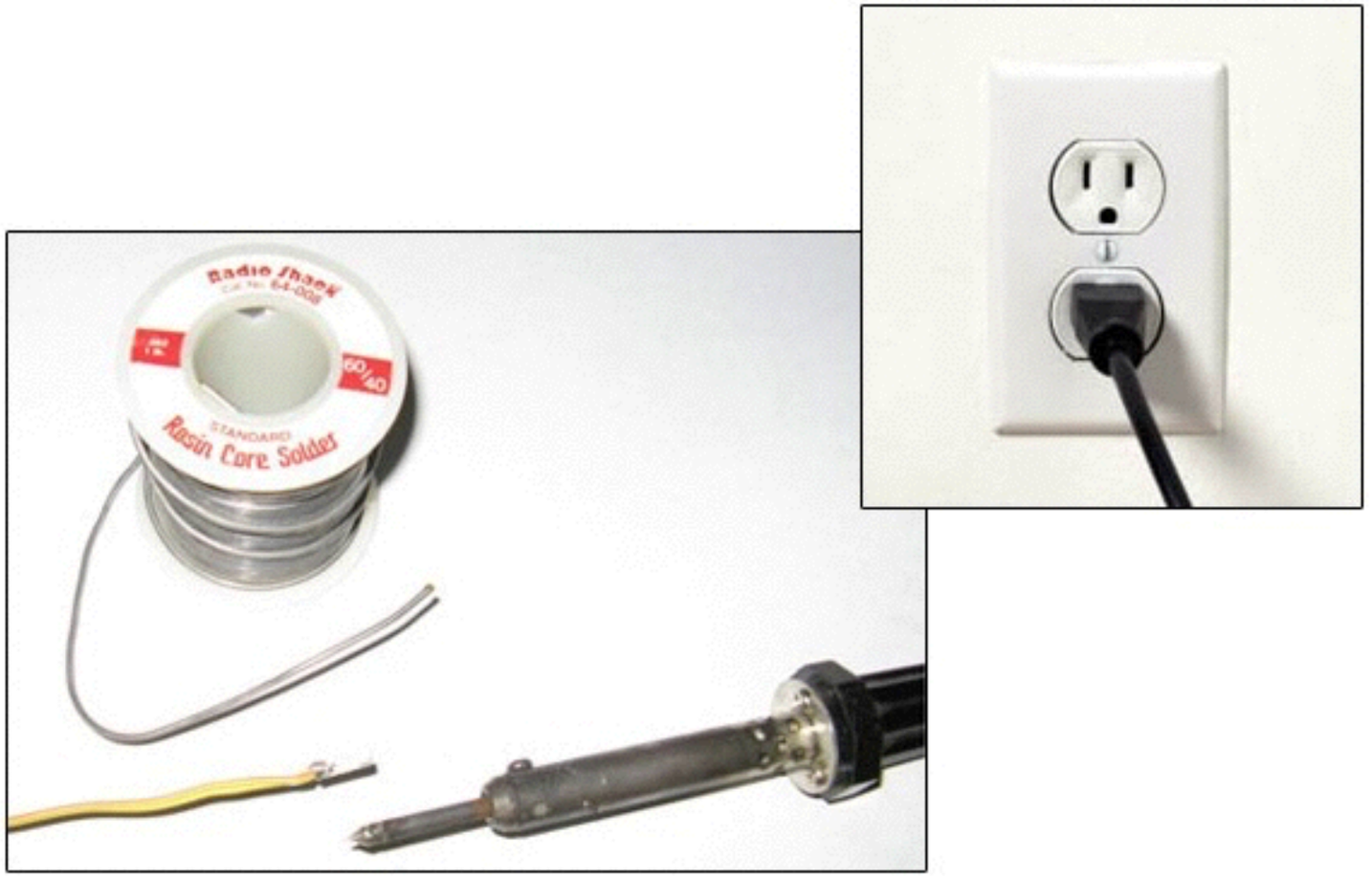
LISKOV SUBSTITUTION PRINCIPLE

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



INTERFACE SEGREGATION PRINCIPLE

You Want Me To Plug This In, Where?



DEPENDENCY INVERSION PRINCIPLE

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