Cheatography

Object Oriented Design Cheat Sheet

by David Harris (david) via cheatography.com/387/cs/179/

SOLID

Single Responsibility Principle
- A class changes for only one reason

Open/Closed Principle
- A class should be open for extension, closed for editing

Liskov's Substitution Principle
- Derived types should cleanly and easily replace base types

Interface Segregation Principle
- Favor multiple single-purpose interfaces over composite

Dependency Inversion Principle
- Concrete classes depend on abstractions, not vice-versa

Other Principles

Don't Repeat Yourself (DRY)
- Duplication should be abstracted

Law of Demeter
- Only talk to related classes

Hollywood Principle
- "Don't call us, we'll call you"

You Ain't Gonna Need It
- Only code what you need now

Keep It Simple, Stupid
- Favor clarity over cleverness

Convention Over Configuration
- Defaults cover 90% of uses

Encapsulation
- What happens in Vegas...

Design By Contract
- And then write tests

Avoid Fragile Base Class
- Treat Base like a public API

Common Closure Principle
- Classes that change together, stay together

Common Refactorings

Encapsulate Field

Generalize Type

Type-Checking ⇒ State/Strategy

Conditional ⇒ Polymorphism

Extract Method

Extract Class

Move/Rename Method or Field

Move to Superclass/Subclass

http://martinfowler.com/refactoring/catalog

Class Associations: Association

Two objects have some sort of relationship to each other.
Example: *Car uses Highway*

Class Associations: Aggregation

An association where one object *has-a* (owns a) different object.
Example: *Car has a Driver*

Class Associations: Composition

An aggregation with dependency - objects are mutually destroyed/created.
Example: *Car has an Engine*

Class Associations: Generalization

"Is-A" relationship (inheritance).
Example: *Porsche is a Car*

Class Associations: Realization

One class implements behavior that is abstractly defined in another class.
Example: *An Animal may Move(), but a Duck would move by waddling*

Class Associations: Dependency

One class weakly depends on another.
Example: *Car uses Highway*

Access Modifiers

Private	Only inside the same class instance
Protected	Inside same or derived class instances
Public	All other classes linking/referencing the class
Internal	Only other classes in the same assembly
Protected Internal	All classes in same assembly, or derived classes in other assembly
Static	Accessible on the class itself (can combine with other accessors)

Design Patterns (GoF)

Abstract Factory	Creational
Builder	Creational
Factory Method	Creational
Prototype	Creational
Singleton	Creational
Adapter	Structural
Bridge	Structural
Composite	Structural
Decorator	Structural
Facade	Structural
Flyweight	Structural
Proxy	Structural
Chain of Responsibility	Behavioral
Command	Behavioral
Interpreter	Behavioral
Iterator	Behavioral
Mediator	Behavioral
Memento	Behavioral
Observer	Behavioral
State	Behavioral
Strategy	Behavioral
Template Method	Behavioral
Visitor	Behavioral

Cheatographer

David Harris (david)
cheatography.com/david/

Cheat Sheet

This cheat sheet was published on 9th January, 2012 and was last updated on 9th January, 2012.

Sponsor