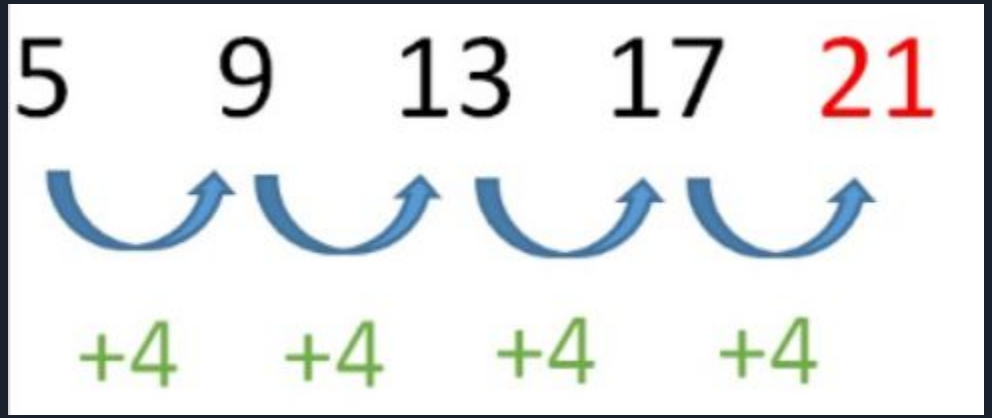
Design patterns

Daniel del Castillo de la Rosa <daniel.del.19@ull.edu.es>
Francisco Jesús Mendes Gomez <francisco.jesus.mendes.gomez.08>

Table of contents

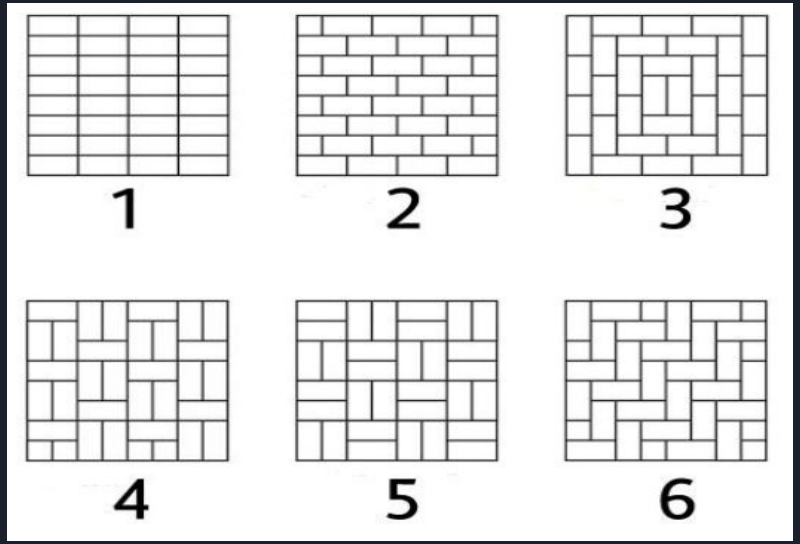
What is a Pattern?

- A pattern are objects which appear recurrently, and we can observe them many times .
- Example :



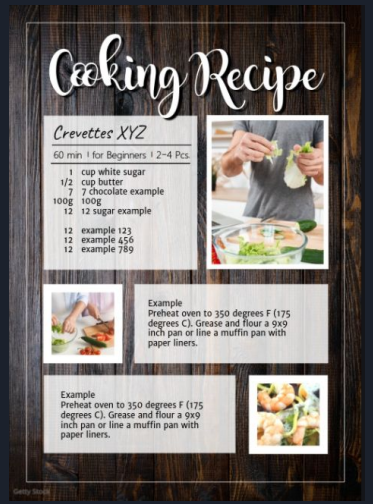
What is a Pattern?

- Another example:



What is a Design Pattern ?

- Definition:
  - Techniques to solve recurring software design problems.



History

- Christopher Alexander, 1979 - The Timeless way of Buildings (Book).

“Each pattern describes a problem that occurs countless times in our environment, as well as the solution to it, in such a way that we can use this solution a million times later without having to think about it again.” - Christopher Alexander.

- Ward Cunningham and Kent Beck, 1987 - Using Patterns Languages for OOP Programs (Article).

- Gang of Four (GoF), Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides - Design Patterns (Book).

Importance of Design Patterns

- Main benefits:
  - Patterns can be expressive.
  - Patterns are proven solutions.
  - Patterns can be easily reused.
- Are considered a best practice on software development.

Structure of Design Patterns.

A design pattern should have a:

- Pattern name and a description
- Context outline
- Problem statement
- Solution
- Design
- Implementation

Structure of Design Patterns.

- Illustrations
- Examples
- Co-requisites
- Relations
- Known usage

Anti-patterns.

- An **anti-pattern** is a common response to a recurring problem that is usually ineffective and risks being highly counterproductive.

- Why should we know about its existence?

Types of Design Patterns

- We will talk about 3 types of Design Patterns:
  - Creational Design Patterns.
  - Structural Design Patterns.
  - Behavioural Design Patterns.

Creational patterns

- RAII
- Builder pattern

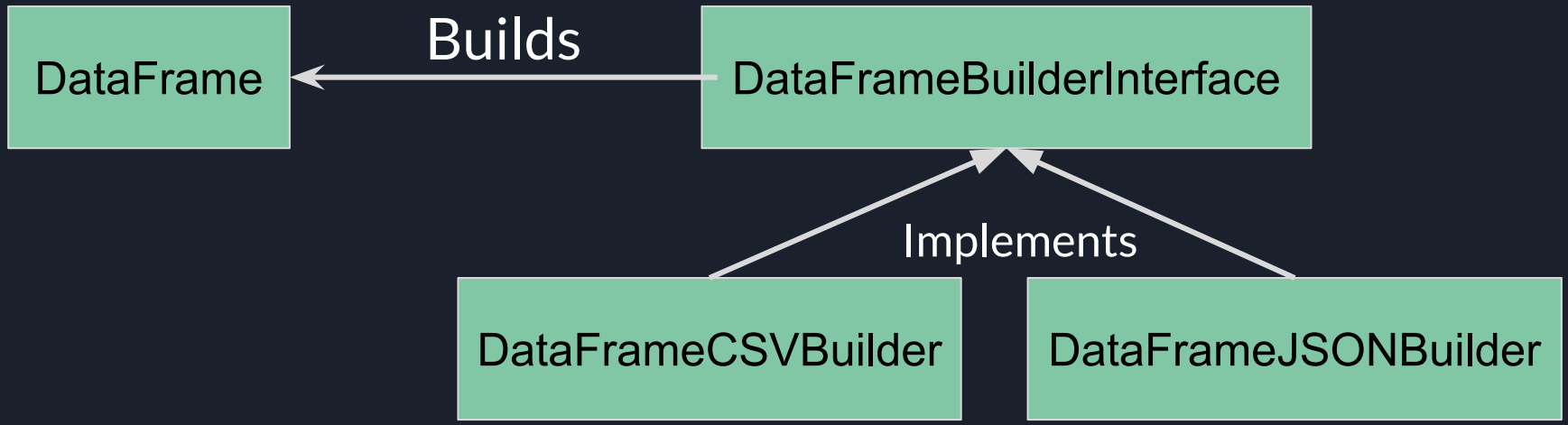
RAII

- Resource Acquisition Is Initialization
- Coined by Bjarne Stroustrup
- Based on use of constructors and destructors
- Encapsulate resource management

Builder pattern

- Simplify classes
- Separate construction of a complex object from its representation

Builder pattern

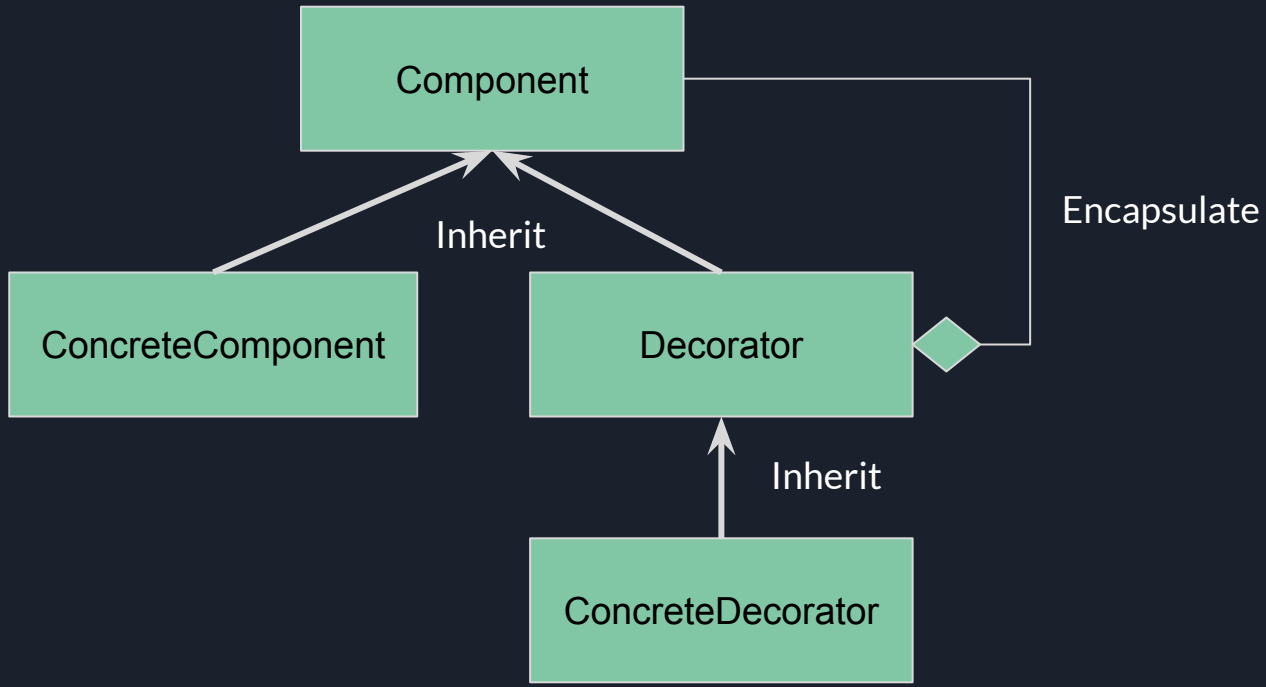


Structural Patterns

- Decorator
  - Main Idea:

    The decorator pattern takes care of adding behavior or functionality to existing classes dynamically.

Decorator Pattern.



Decorator Pattern.

Advantages:

- We can extend the behavior of an object without creating new subclasses.
- Allow us add or extract behaviours in execution time.
- We can use different decorators to encapsulate an object.
- We can use decorated objects as components.

Behavioural patterns

- Iterator pattern
- Observer pattern
- Chain of responsibility pattern

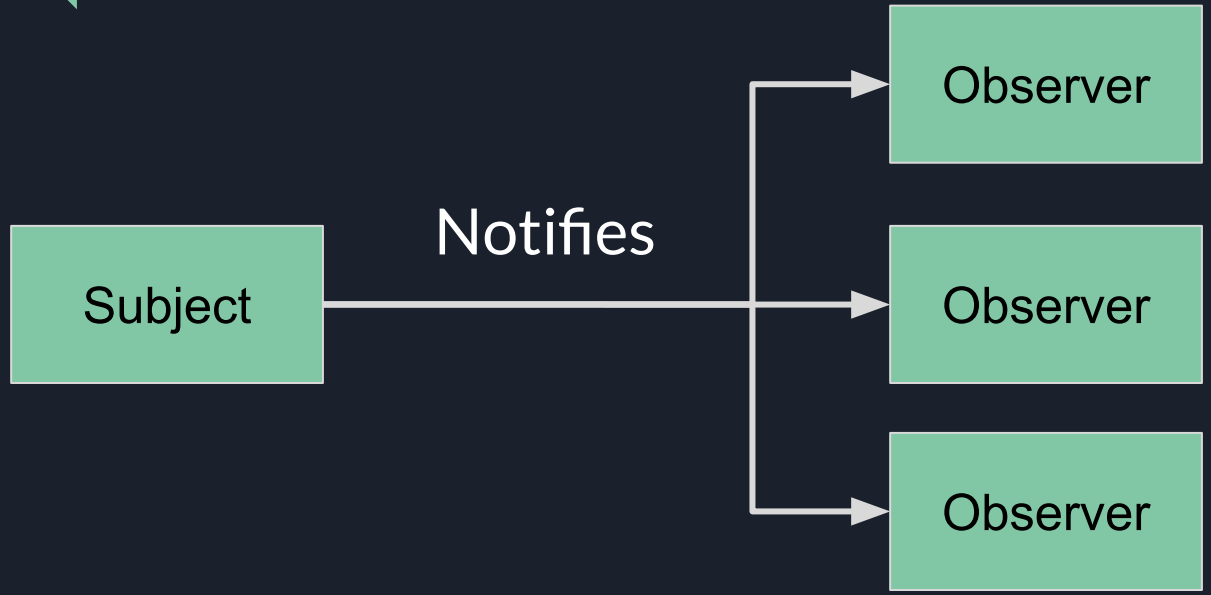
Iterator pattern

- Traverse a container
- Abstract from container type

Observer pattern

- Observers wait for an event
- Another class notifies events to observers
- The observers perform an action when notified
- One-to-many dependency without tight coupling

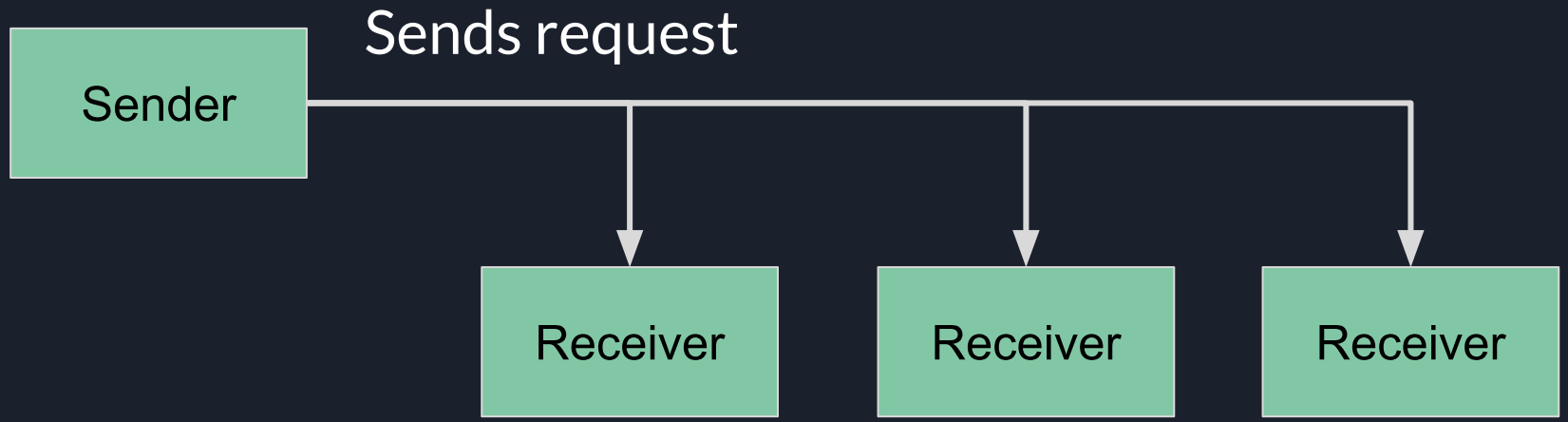
Observer pattern



Chain of responsibility pattern

- Composed of commands and processing objects
- Each command gets passed until it finds an object that can process it
- Avoid tight coupling between senders and receivers

Chain of responsibility pattern



Conclusions

- What is a design pattern
- Types of design pattern
  - Creational
  - Structural
  - Behavioural

Bibliography

- Wikipedia
- Learning JavaScript Design Patterns by Addy Osmani
- JavaScript Design Patterns
- Rust unofficial
- JavaScript - Patrones de diseño en JS

Thanks.