Flyweight

Karoly Nyisztor
DEVELOPER

@knyisztor   www.leakka.com

Overview

Motivation

FlyweightFactory

- Naive approach

- Apply the Flyweight

Motivation

Share one instance to represent many

- Separate state-dependent part
- Store intrinsic/immutable state in the Flyweight object
- Extrinsic state managed by clients
- Callers should be able to modify individual state, not the intrinsic one

Flyweight

The Flyweight pattern shares common objects that can be used identically.

Demo

FlyweightFactory Demo

- The problem
- Flyweight refactoring effects

Summary

Use the Flyweight to:

- Share one instance to represent many
- Separate state-dependent and state-independent part
- Encapsulate immutable state in the Flyweight object
- Allow clients to manage the extrinsic state managed by clients

Pitfalls

- Allow changes of the immutable state
- Not protecting against concurrent access