Basic Creational Patterns

Zachary Bennett
Software Engineer

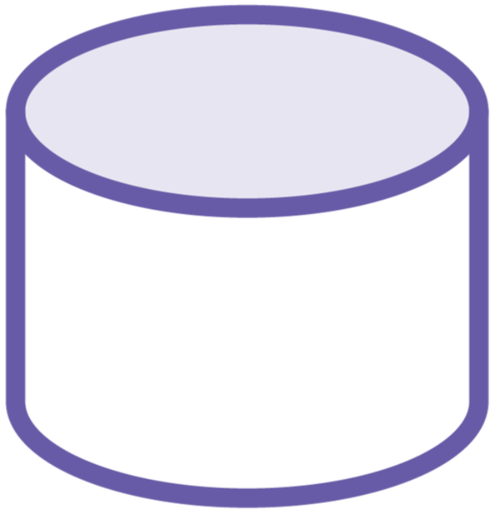
@z_bennett_ zachbennettcodes.com

Creational Design Pattern

An object-oriented design pattern that focuses on improving object instantiation.

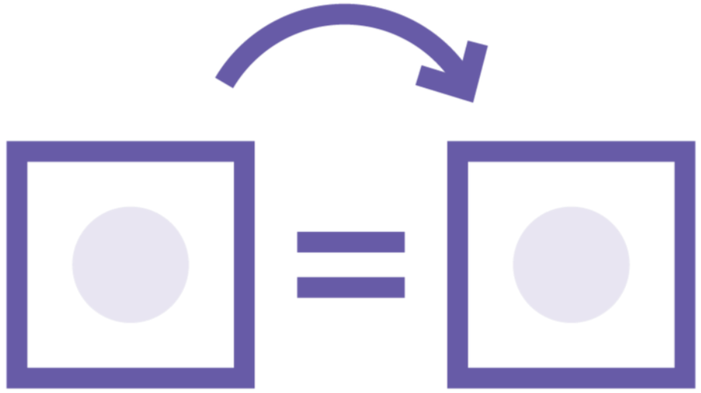
Creational patterns are all about improving how you create objects

Basic Creational Patterns



Singleton
Sharing state

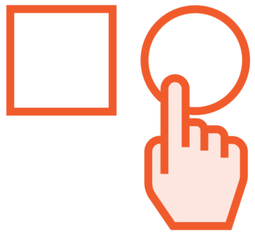
Builder
Simplifying complex object creation



Prototype
Cloning objects

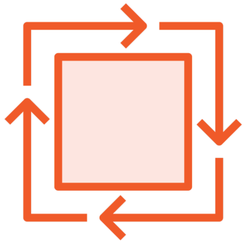
Singleton Pattern

Why the Singleton Pattern?



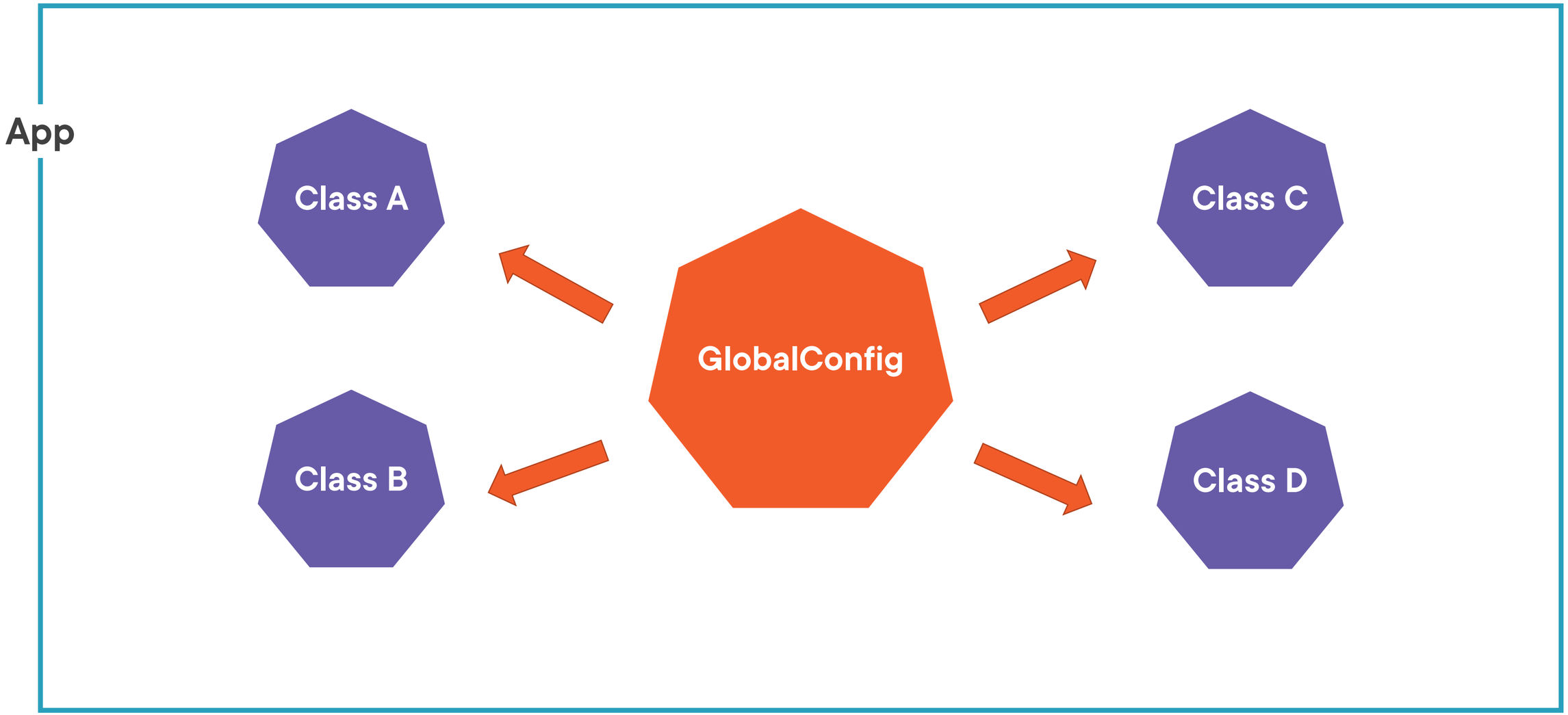
You want to ensure that an object is only instantiated once

You want to use global state in your program or app

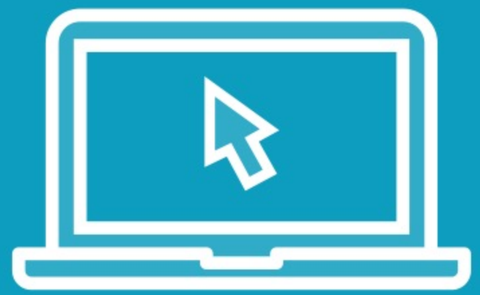


You want to provide a central location for shared state

The Singleton Pattern



Demo

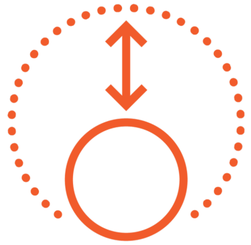


Singleton Design Pattern

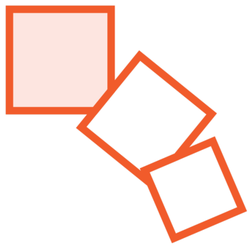
- GlobalCoffeeConfig class
- Example usage

Builder Pattern

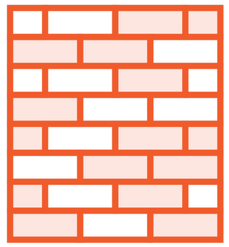
Why the Builder Pattern?



Allow customization of object construction



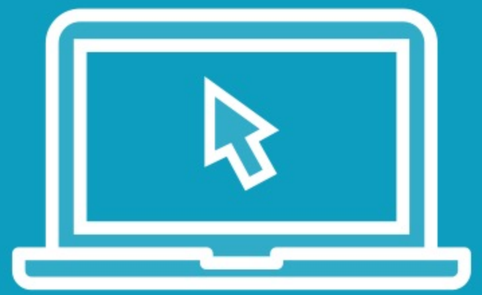
Provide a declarative, step-by-step API for object creation



Simplify object construction

The Builder Pattern

Demo

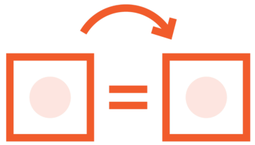


Builder Design Pattern

- Coffee class implementation
- Build coffee objects step-by-step

Prototype Pattern

Why the Prototype Pattern?

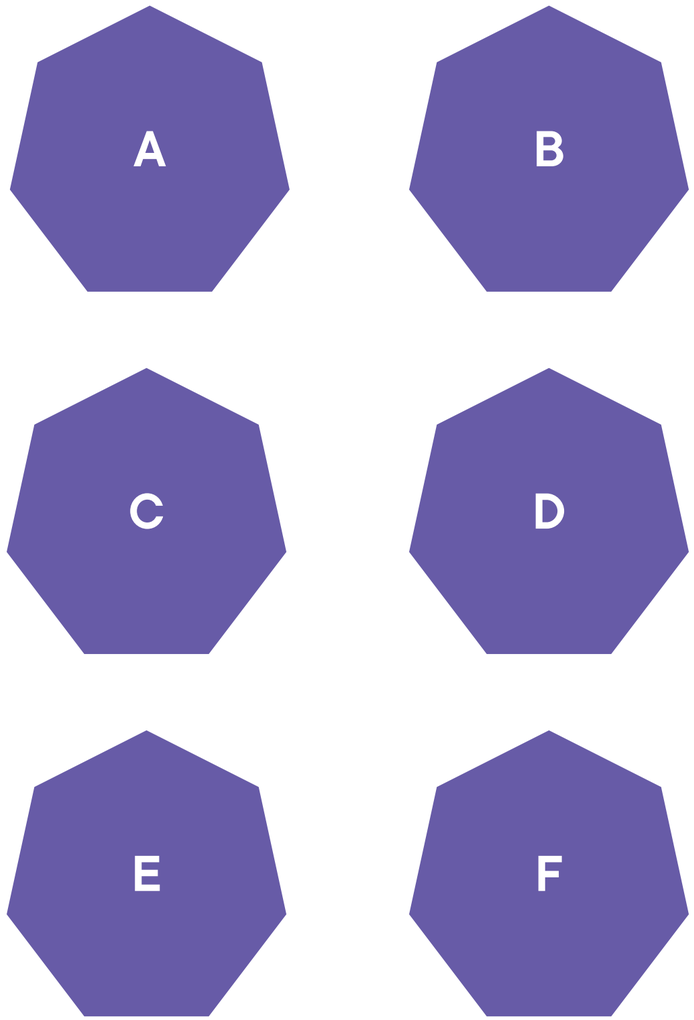


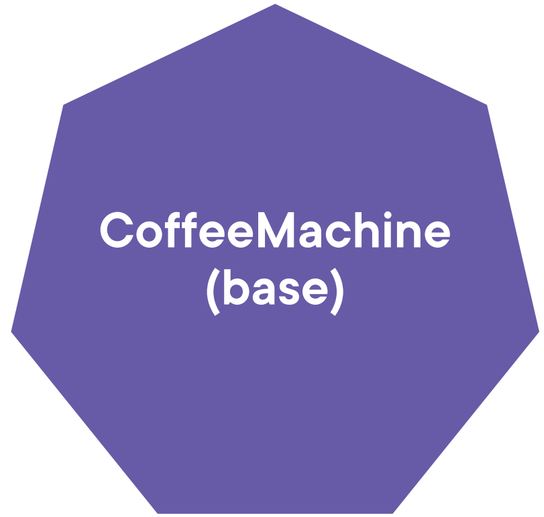
You need to clone/copy large objects

You are constantly needing to reconfigure created objects

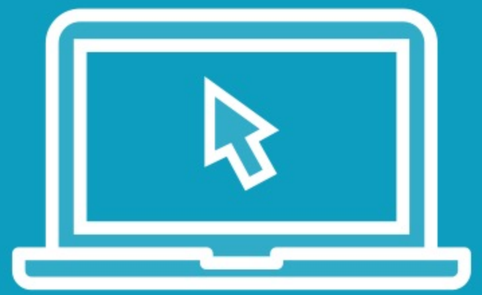
Dynamic object creation

The Prototype Pattern





Demo



Prototype Design Pattern

- Prototypal coffee machine class
- Usage and object cloning

Summary

Creational design patterns

Aiding with object creation

Basic patterns

- Singleton
- Builder
- Prototype