Singleton

BRIAN LAMARCHE

COMPUTER SCIENCE 323 – SOFTWARE DESIGN

Singleton

When:

- …You want to only create a single instance of an object
- …You need to coordinate actions across a system

Examples:

- File systems
- Window Managers
- Resources
- Hardware

Singleton

Creational Design Pattern

This pattern relates to/with:

- Abstract Factory
- Factory
- Builder

Used Well With

- Disposable

Motivation:

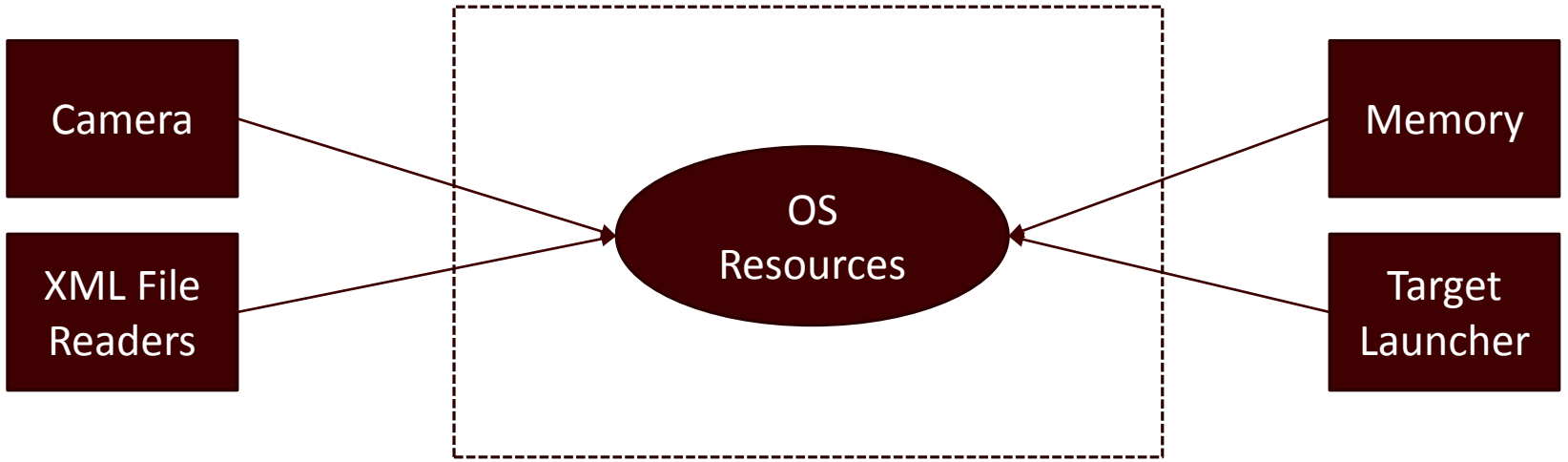
- You need only one instance of an object with accessibility from clients globally across the system

Example: Resource Manager

Previously, we discussed IDisposable and interfaces, a resource manager may be an object you want to create that manages all resources in your program, during the life of your program.

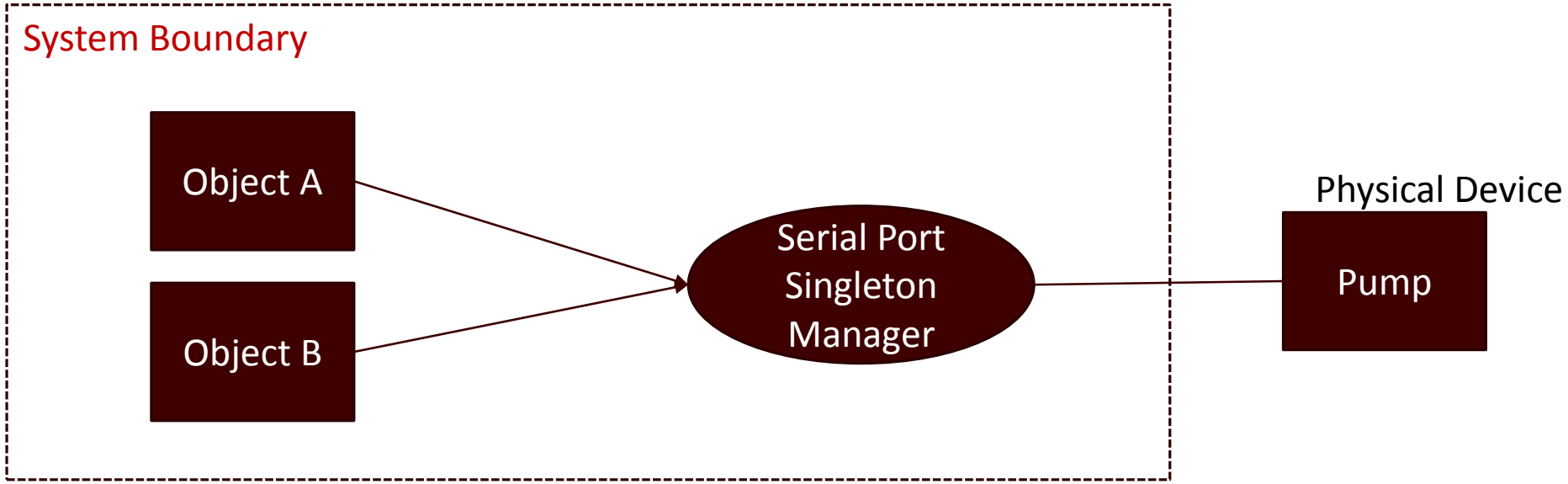
This is a perfect example of when to use a singleton.

Example: Singleton & Design Pattern



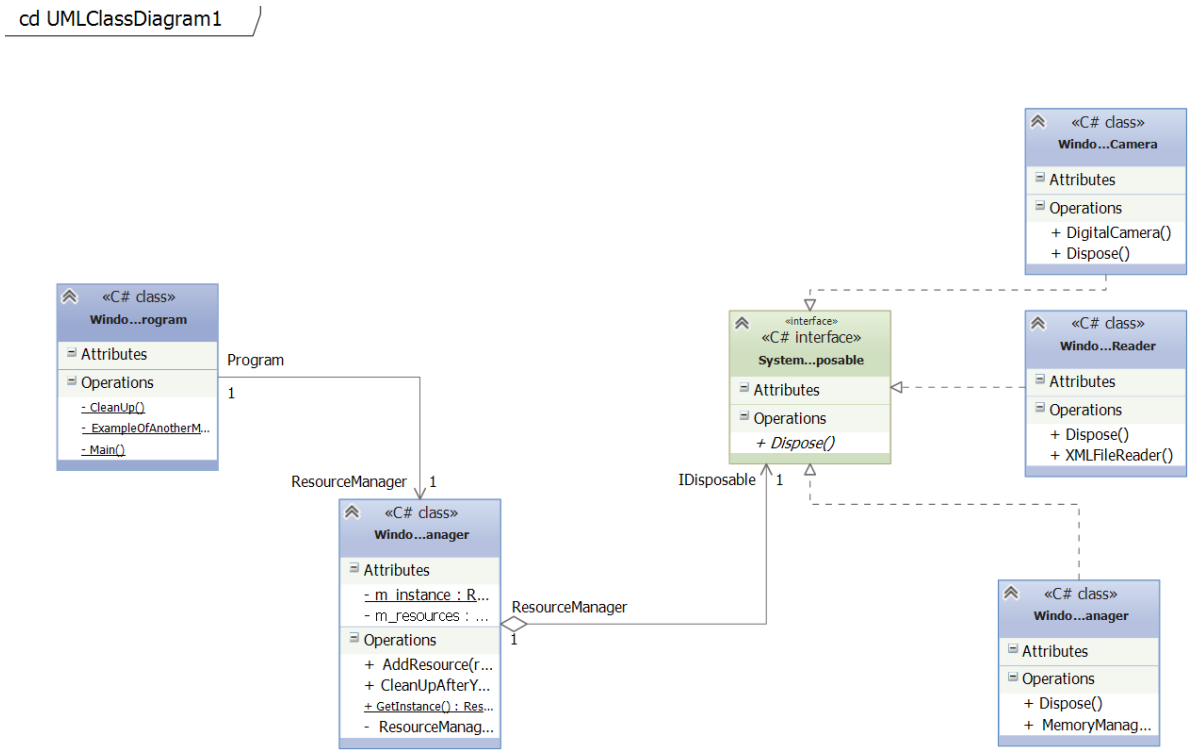
Also use a Singleton design pattern to manage the resources.

Example: Singleton & Design Pattern



Also use a Singleton design pattern to manage the resources.

UML Modeling:



Example Code:

```
public class ResourceManager
 {
    /// <summary>
    /// Singleton reference.
    /// </summary>
    private static ResourceManager m_instance;

    /// <summary>
    /// Lazy implementation that gets the instance of the singleton.
    /// </summary>
    /// <returns></returns>
    public static ResourceManager GetInstance()
    {
            if (m_instance == null)
            {
                    m_instance = new ResourceManager();
            }
             return m_instance;
    }

    /// <summary>
    /// Constructor.
    /// </summary>
    private ResourceManager()
    {
        m_resources = new List<IDisposable>();
    }
            …
```

Pros

Controlled Access

- You can control over how and when it's accessed

Control refinements

- If sub-classed then you can control what the application receives at run time. i.e. you control how it's built and what is built.

Permit multiple instances

- You can allow for multiple instances to be created through the accessor.
  - E.g. You allow for an internet connection and only allow four access instances to be created for four available ports.

Cons

Synchronization

- If you are using multiple threads, consider safe access to those objects