Design Patterns

Singleton

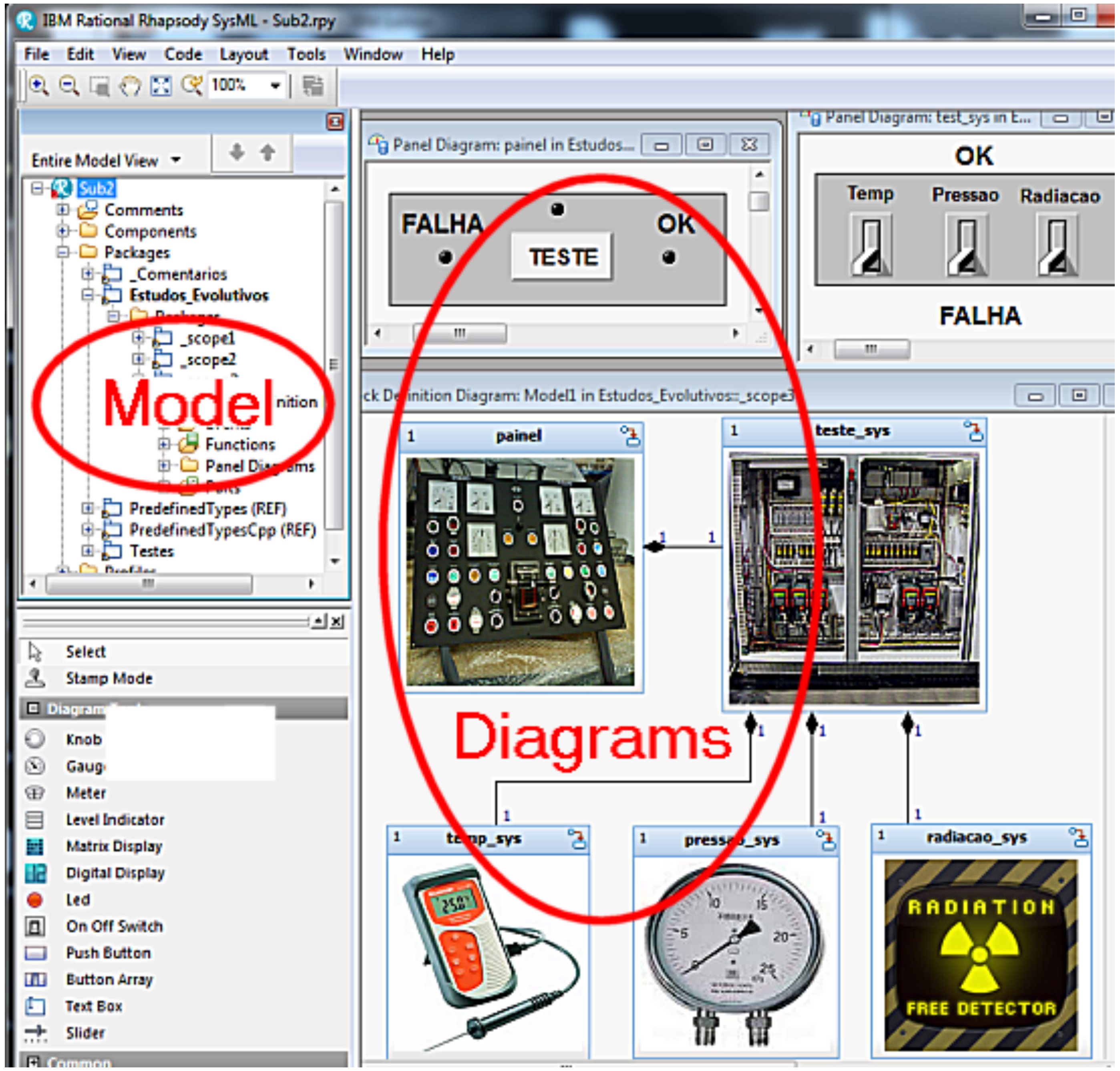
COMP3607
Object Oriented Programming II

Week 5

Outline

- Drawing vs Modelling
- Singleton Design Pattern
- Composite Design Pattern

Drawing vs Modelling



https://www.ibm.com/developerworks/community/blogs/invisiblethread/entry/
a_diagram_is_not_a_model_the_huge_difference_between_them?lang=en_us

1 Singleton Design Pattern

The Singleton design pattern is used to ensure that a class has only one instance, and to provide a global point of access to it.

Reading Resource: Design Patterns, Chapter 3, pp 127-134

1

Problem Scenario

Consider a scenario where a system has many printers, but there should be only one printer spooler*.

How do we ensure that the printer spooler class has only **one** instance of a printer spooler object, and that the instance is **easily accessible**?

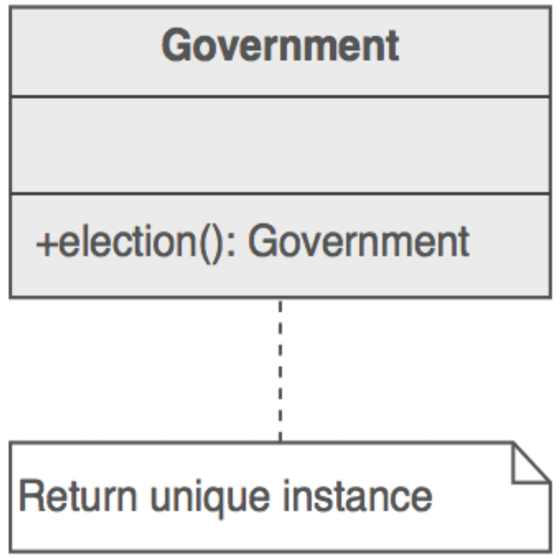
Possible options:

- A global variable - problem with this?
- Make the class responsible for keeping track of the sole instance and for giving access to it.

* software responsible for managing all print jobs by adding and removing jobs from a queue.

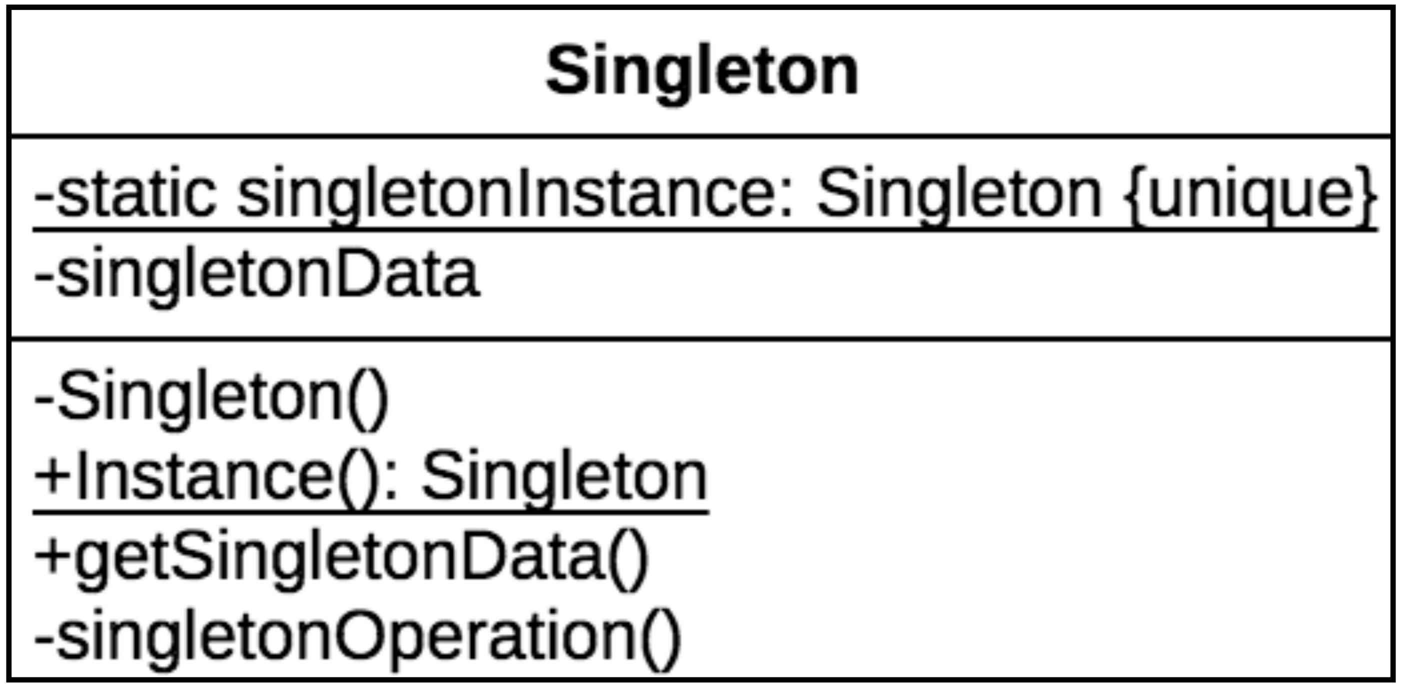
Another Example

The Singleton pattern ensures that a class has only one instance and provides a global point of access to that instance. It is named after the singleton set, which is defined to be a set containing one element. The office of the President of the United States is a Singleton. The United States Constitution specifies the means by which a president is elected, limits the term of office, and defines the order of succession. As a result, there can be at most one active president at any given time. Regardless of the personal identity of the active president, the title, "The President of the United States" is a global point of access that identifies the person in the office.



https://sourcemaking.com/design_patterns/singleton

Singleton Design Pattern (UML)



Code Example

```
public class Singleton {

    private static Singleton singletonInstance;//unique instance
    private String singletonData;

    private Singleton() {// private constructor
       singletonData = “A single tonne of data is a lot of data”;
    }

    public static Singleton Instance() { //class method to get to unique instance
      if(singletonInstance == null)
          singletonInstance = new Singleton();
        return singletonInstance;
    }

    public String getSingletonData() {
        return singletonData;
    }

    private void singletonOperation() {
        // TODO implement other stuff here
    }

}
```

Applicability: Singleton

The Singleton pattern is used when:

- There must be exactly one instance of class, and it must be accessible to clients from well-known access point.
- The sole instance should be extensible by subclassing, and clients should be able to use an extended instance without modifying their code.

1

Consequences: Singleton

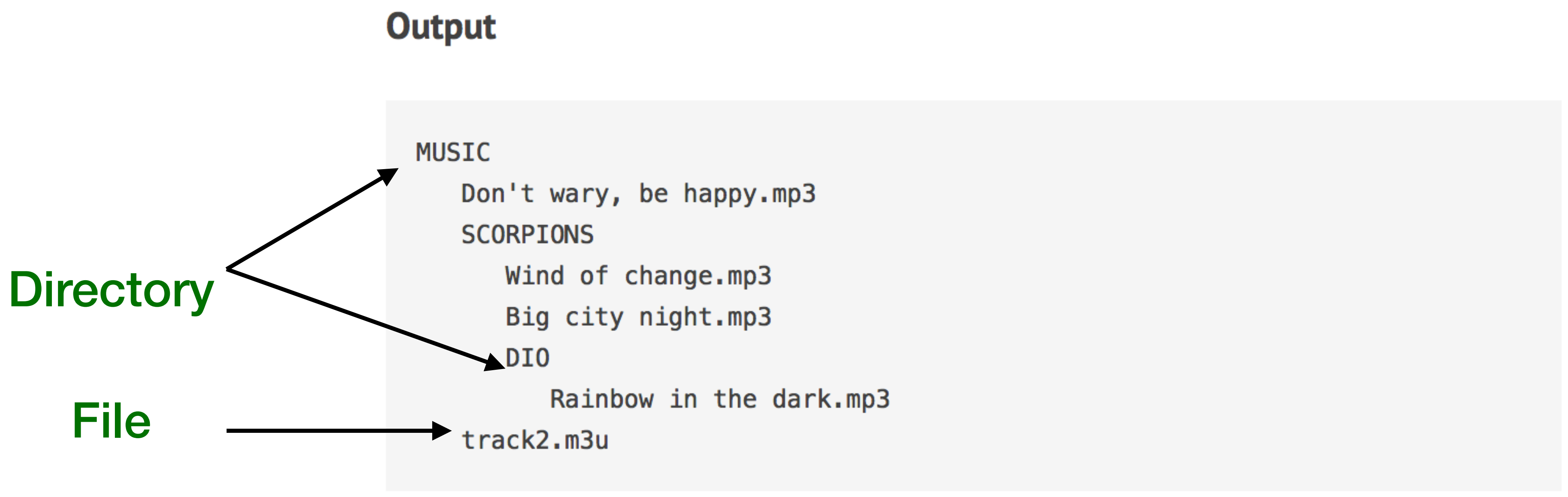
Benefits:

- Controlled access to sole instance: (encapsulation)
- Reduced name space: (less global variables)
- Refinement of operations: (subclassing)
- Variable number of instances: (controlled by class)
- More flexible than class operations: (language quirks)

1

Exercise

Design a solution for listing all of the files and directories stored on a disk. Output should be presented as:



Output

```
MUSIC
   Don't wary, be happy.mp3
   SCORPIONS
      Wind of change.mp3
      Big city night.mp3
      DIO
         Rainbow in the dark.mp3
   track2.m3u
```

2

https://sourcemaking.com/design_patterns/composite/java/1

References

- UML: online reading resources
  - www.uml.org (Documentation, notation)
  - www.omg.org (Standards, protocols)
- UML Modelling and Drawing Tool:
  - http://staruml.io/