Memento

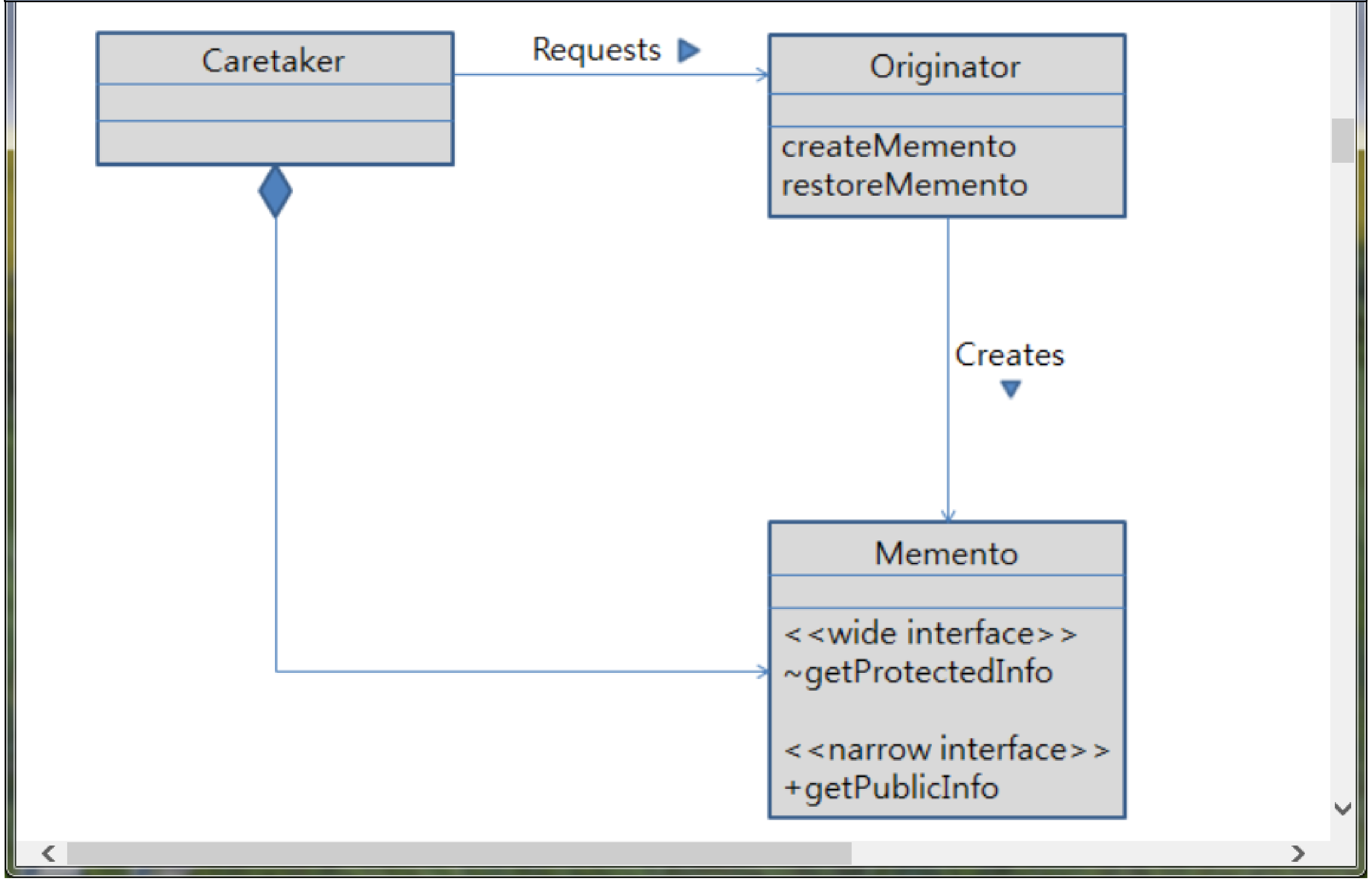
2022, Spring

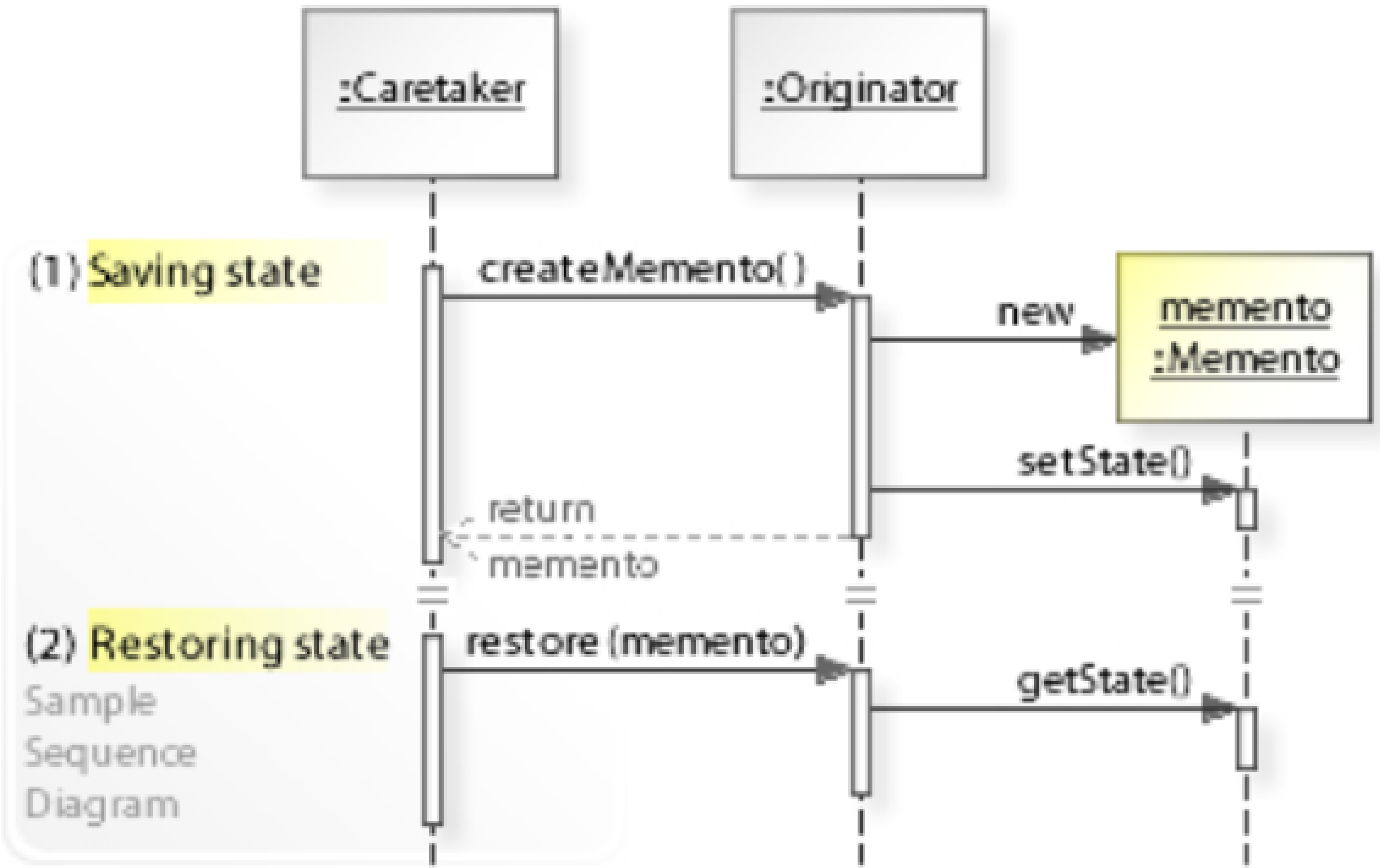
The memento pattern is a software design pattern that provides the ability to restore an object to its previous state (undo via rollback).

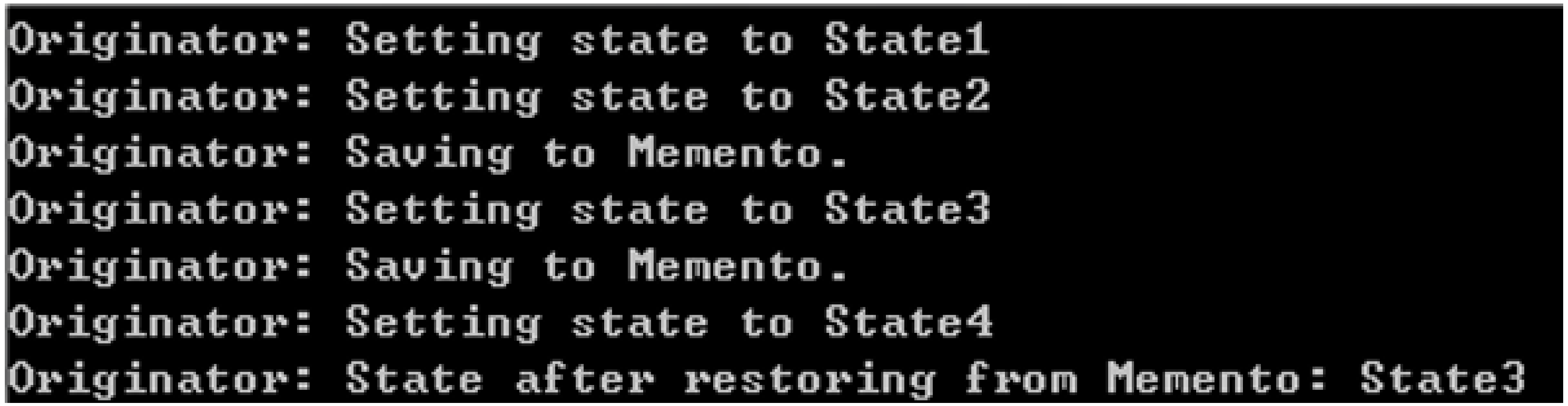
The **originator** is some object that has an **internal state**.
The **caretaker** is going to **do something to the originator**, and wants **to be able to undo the change**.

The **memento object** itself is an opaque object (one which the **caretaker cannot**, or should not, change)

Memento


Caretaker
Requests ▶
Originator
createMemento
restoreMemento
Creates ▼
Memento
<<wide interface>>
~getProtectedInfo
<<narrow interface>>
+getPublicInfo


:Caretaker
:Originator
(1) Saving state
createMemento( )
new
memento
:Memento
setState()
return
memento
(2) Restoring state
restore (memento)
getState()
Sample
Sequence
Diagram



```
Originator: Setting state to State1
Originator: Setting state to State2
Originator: Saving to Memento.
Originator: Setting state to State3
Originator: Saving to Memento.
Originator: Setting state to State4
Originator: State after restoring from Memento: State3
```

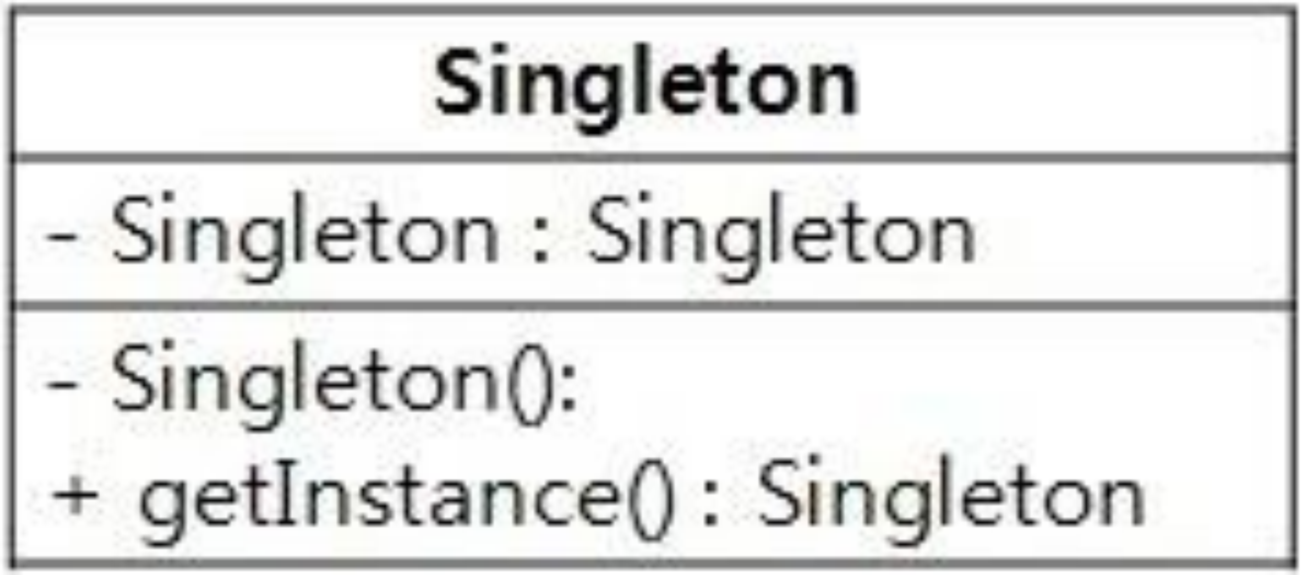
```
import java.util.*;
class Caretaker {
    public static void main(String[] args) {
        List<Originator.Memento> savedStates =
                new ArrayList<Originator.Memento>();
        Originator originator = new Originator();
        originator.set("State1");
        originator.set("State2");
        savedStates.add(originator.saveToMemento());
        originator.set("State3");
       savedStates.add(originator.saveToMemento());
        originator.set("State4");
        originator.restoreFromMemento(savedStates.get(1));
  }
}
```

```
import java.util.List;
import java.util.ArrayList;
class Originator {
    private String state;
    public void set(String state) {
        System.out.println("Originator: Setting state to " + state);
        this.state = state;     }
    public Memento saveToMemento() {
        System.out.println("Originator: Saving to Memento.");
        return new Memento(state);     }
    public void restoreFromMemento(Memento memento) {
        state = memento.getSavedState();
        System.out.println("Originator: State after restoring from
                  Memento: ” + state);
    }
```

```
 public static class Memento {
        private final String state;
        private Memento(String stateToSave) {
            state = stateToSave;
        }
        private String getSavedState() {
            return state;
        }
  }
} // end of Originator
```

Singleton Pattern

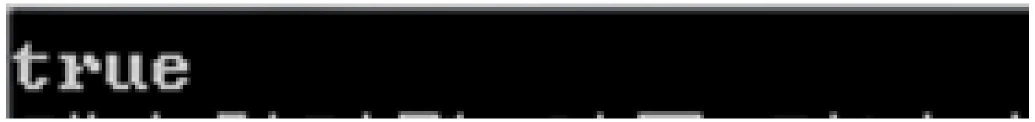
Compare Singleton with Memento



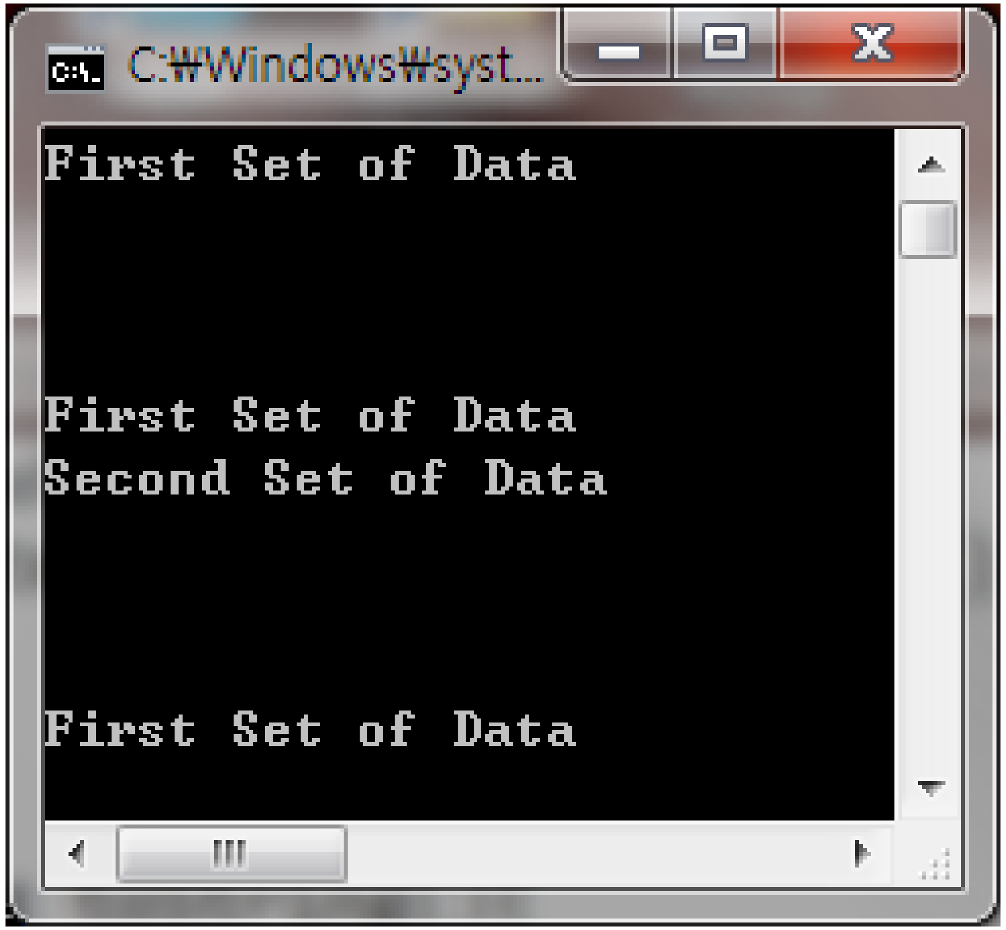
```java
import java.util.*;
public class MySingleton {
    //the static singleton object
    private static  MySingleton theObject;
    private MySingleton() {
    }
    public static MySingleton createMySingleton() {
        if (theObject == null)
            theObject = new MySingleton();
        return theObject;
    }

}
```

```
public class Main {
        public void createSingleton() {
        MySingleton ms1 =
         MySingleton.createMySingleton();
        MySingleton ms2 =
         MySingleton.createMySingleton();
        System.out.println( ms1 == ms2 );
    }
    public static void main(String[] args) {
      new  Main().createSingleton();
    }
}
```



```
true
```



```
First Set of Data



First Set of Data
Second Set of Data



First Set of Data
```

```
public class FileWriterUtil {
    private String fileName;
    private StringBuilder content;
    public FileWriterUtil(String file){
        this.fileName=file;
        this.content=new StringBuilder();
    }
    @Override
    public String toString(){
        return this.content. toString();
    }
    public void write(String str){
        content. append(str);
    }
public Memento save(){
        return new Memento(this.fileName, this.content);
    }
    public void undoToLastSave(Object obj){
        Memento memento = (Memento) obj;
        this.fileName= memento.fileName;
        this.content=memento.content;
    }
    private class Memento{
        private String fileName;
        private StringBuilder content;

        public Memento(String file, StringBuilder content){
            this.fileName=file;
            this.content=new StringBuilder(content);
        }
    }  }
```

```
public class FileWriterCaretaker {
    private Object obj;
    public void save(FileWriterUtil fileWriter){
        this.obj=fileWriter.save();
    }
    public void undo(FileWriterUtil fileWriter){
        fileWriter.undoToLastSave(obj);
    }
}
```

```
public class FileWriterClient {
    public static void main(String[] args) {
        FileWriterCaretaker caretaker = new FileWriterCaretaker();
        FileWriterUtil fileWriter = new FileWriterUtil("data.txt");
        fileWriter.write("First Set of Data₩n");
        System.out.println(fileWriter+"₩n₩n");
        // lets save the file
        caretaker.save(fileWriter);
        //now write something else
        fileWriter.write("Second Set of Data₩n");
        //checking file contents
        System.out.println(fileWriter+"₩n₩n");
        //lets undo to last save
        caretaker.undo(fileWriter);
        //checking file content again
        System.out.println(fileWriter+"₩n₩n");
    }
}
```