Memento Pattern

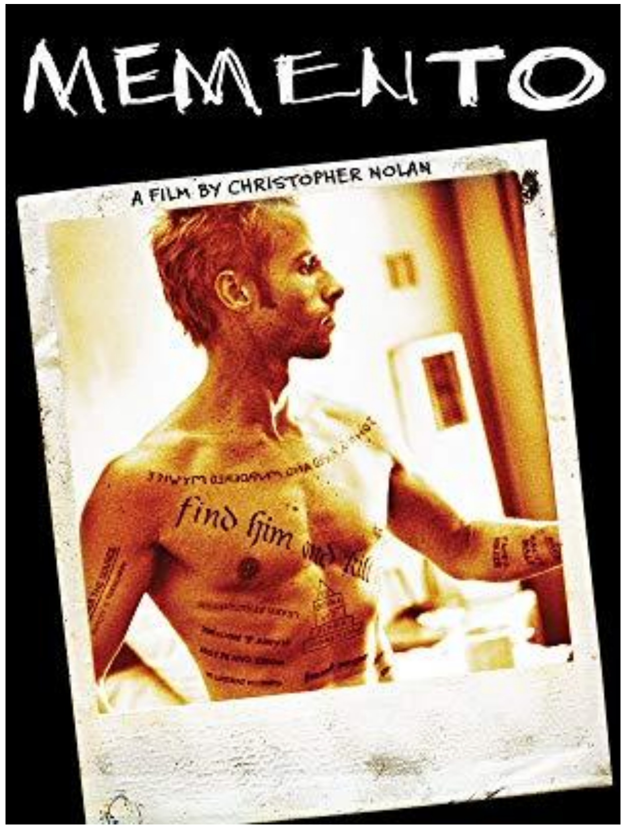
Vibhu A Bharadwaj, Quan Liang, Jim Fawcett
CSE-776 Design Patterns, Fall 2018
Syracuse University

Intent and Motivation

- Provide a mechanism to save and later restore the internal state of an object without violating encapsulation.

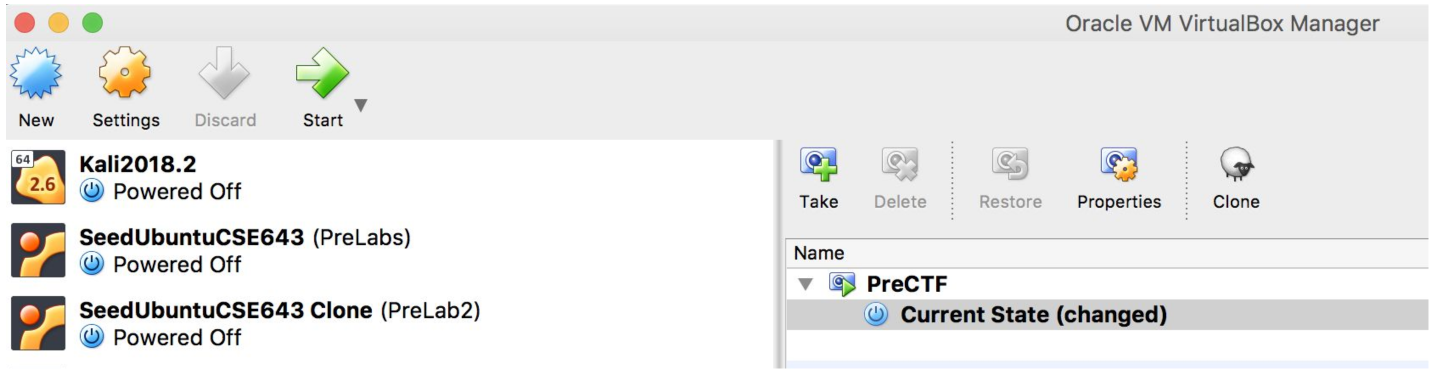
- Adaptation to change while maintaining the inherent stateful-ness.

- The safety net to fall back on!



1. https://images-na.ssl-images-amazon.com/images/I/8168mMWsxHL._RI_SX300_.jpg
2. https://forums.nextgames.com/walkingdead/discussion/33100/can-we-get-an-undo-button

Participants

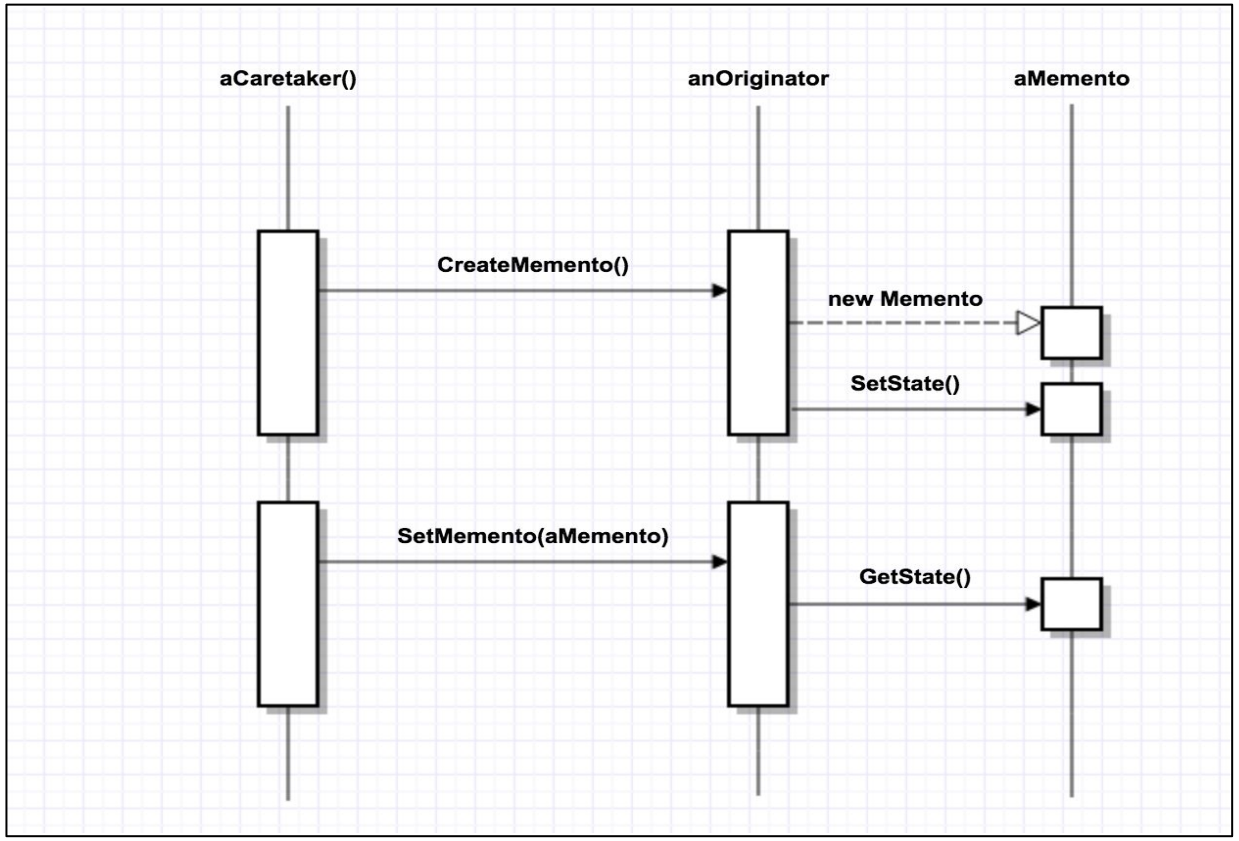


Oracle VM VirtualBox Manager

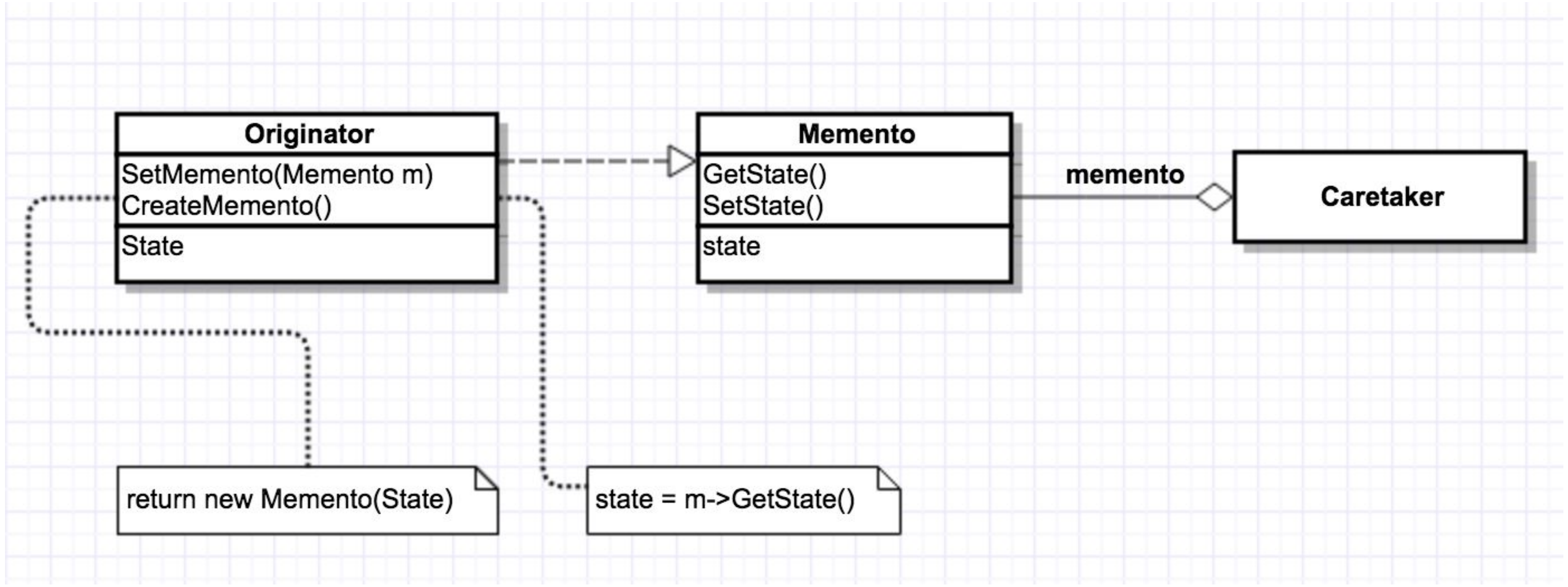
Kali2018.2
Powered Off
SeedUbuntuCSE643 (PreLabs)
Powered Off
SeedUbuntuCSE643 Clone (PreLab2)
Powered Off

PreCTF
Current State (changed)

- Caretaker - Oracle VM VirtualBox provides the mechanism to take a snapshot
- Originator - The state of my VM which I want to take a snapshot of.
- Memento - The snapshot of my VM which stores all the information necessary to restore to my previous snapshot



Applicability and Structure



Design Patterns, Elements of Reusable Object-Oriented Software, Erich Gamma, et. al.

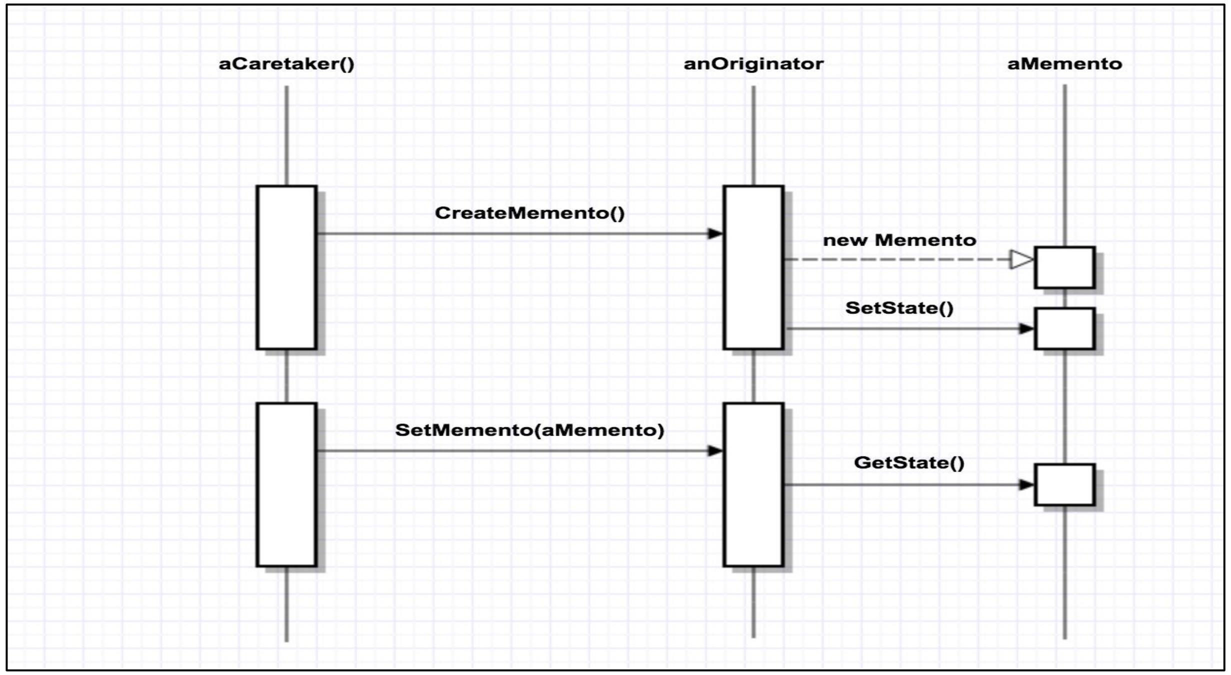
- The Narrow Interface: Memento and Caretaker
- The Wide Interface: Memento and Originator
- The Memento stores just enough state information to revert to/retrieve/recreate the original state accurately
- Maintain encapsulation boundaries

Collaborations

- Caretaker asks for a memento or snapshot or token to be created by originator.

- Caretaker holds the memento and returns it to Originator if needed (not necessarily returned)

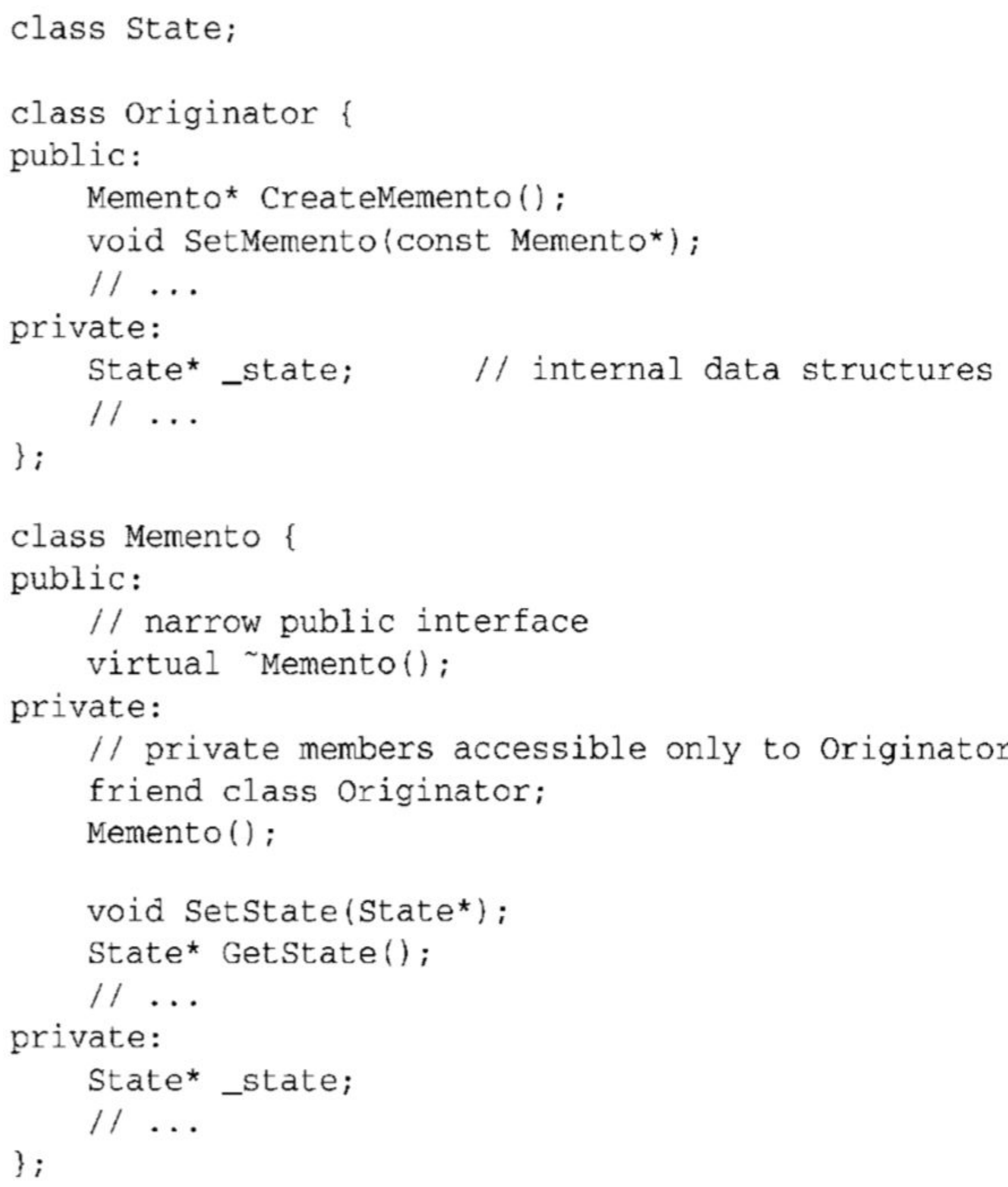
- Mementos are passive, the originator owns the memento and only the originator should be able to assign/retrieve state.



Consequences

- Preserving encapsulation boundaries
- Simplifies the originator

- Using mementos might be expensive.
- Hidden costs in caring for mementos



```
class State;

class Originator {
public:
    Memento* CreateMemento();
    void SetMemento(const Memento*);
    // ...
private:
    State* _state;      // internal data structures
    // ...
};

class Memento {
public:
    // narrow public interface
    virtual ~Memento();
private:
    // private members accessible only to Originato
    friend class Originator;
    Memento();

    void SetState(State*);
    State* GetState();
    // ...
private:
    State* _state;
    // ...
};
```

Implementation issue

- Language support for wide and narrow interface
- Storing incremental changes

Known Uses

- Whenever you need to roll back the state of the object
- Undo command like ctrl-z in file editor
- Database transaction operation
- Save and reload the progress when playing a game
- Cookies for retrieving session information

Related Patterns

- Iterator - Mementos objects can be used to represent the iteration state (the collection is a friend of an IteratorState)
- Command + Memento -> Maintain state of undoable operations

References

1. https://forums.nextgames.com/walkingdead/discussion/33100/can-we-get-an-undo-button
2. https://images-na.ssl-images-amazon.com/images/I/8168mMWsxHL._RI_SX300_.jpg
3. Design Patterns, Elements of Reusable Object-Oriented Software, Erich Gamma, et. al.