COMMAND PATTERN

DANIEL NAEL
COMMAND PATTERN

INTENT

Encapsulate a request as an object, thereby letting users parameterize clients with different requests, queue or log requests, and support undoable operations.

- Design Patterns: Elements of Reusable Object-Oriented Software, p263
COMMAND PATTERN

INTENT

Encapsulate a request as an object, thereby letting users parameterize clients with different requests, queue or log requests, and support undoable operations.

- Design Patterns: Elements of Reusable Object-Oriented Software, p263
COMMAND PATTERN

INTENT

Encapsulate a request as an object, thereby letting users parameterize clients with different requests, queue or log requests, and support undoable operations.

IN SIMPLER WORDS
COMMAND PATTERN

INTENT

Encapsulate a request as an object, thereby letting users parameterize clients with different requests, queue or log requests, and support undoable operations.

IN SIMPLER WORDS

Turn an ACTION into DATA.
COMMAND PATTERN

CLIENT

INVOKER

Command
  + Execute()

ConcreteCommand
  + Execute()

Receiver
  + Action()
COMMAND PATTERN

CHARACTERISTICS

Function calls become:

- Easy to store
- Easy to reuse
- Delayed if necessary
- Configurable during runtime
CHARACTERISTICS

Function calls become:

- Easy to store
- Easy to reuse
- Delayed if necessary
- Configurable during runtime
CHARACTERISTICS

Function calls become:

- Easy to store
- **Easy to reuse**
- Delayed if necessary
- Configurable during runtime
COMMAND PATTERN

CHARACTERISTICS

Function calls become:

- Easy to store
- Easy to reuse
- **Delayed if necessary**
- Configurable during runtime
COMMAND PATTERN

CHARACTERISTICS

Function calls become:

- Easy to store
- Easy to reuse
- Delayed if necessary
- **Configurable during runtime**
POTENTIAL USES

- Store, queue, reuse, log and execute requests at different times.
- Great for managing input (rebinding)
- Decouple in game entities from players/AI
- Support undo/redo
- Support replays
- Support time rewind mechanics
COMMAND PATTERN

POTENTIAL USES

- Store, queue, reuse, log and execute requests at different times.
- Great for managing input (rebinding)
- Decouple in game entities from players/AI
- Support undo/redo
- Support replays
- Support time rewind mechanics
**COMMAND PATTERN**

**POTENTIAL USES**

- Store, queue, reuse, log and execute requests at different times.
- **Great for managing input (rebinding)**
- Decouple in game entities from players/AI
- Support undo/redo
- Support replays
- Support time rewind mechanics
COMMAND PATTERN

POTENTIAL USES

- Store, queue, reuse, log and execute requests at different times.
- Great for managing input (rebinding)
- **Decouple in game entities from players/AI**
- Support undo/redo
- Support replays
- Support time rewind mechanics
POTENTIAL USES

- Store, queue, reuse, log and execute requests at different times.
- Great for managing input (rebinding)
- Decouple in game entities from players/AI
- **Support undo/redo**
- Support replays
- Support time rewind mechanics
POTENTIAL USES

- Store, queue, reuse, log and execute requests at different times.
- Great for managing input (rebinding)
- Decouple in game entities from players/AI
- Support undo/redo
- **Support replays**
- Support time rewind mechanics
POTENTIAL USES

- Store, queue, reuse, log and execute requests at different times.
- Great for managing input (rebinding)
- Decouple in game entities from players/AI
- Support undo/redo
- Support replays
- **Support time rewind mechanics**
EXAMPLE

Icarus - A Real-Time Strategy Game
EXAMPLE

AI PLAYER

Sets unit state

Keyboard and mouse player

Sets unit state

Networked Player

Sets unit state

Units
For the next week …

- Read Command chapter from Game Programming Patterns [http://www.gameprogrammingpatterns.com/command.html](http://www.gameprogrammingpatterns.com/command.html)
- Complete tasks **MoveCommand** and **SwitchCommand**