Command and State Patterns

Curt Clifton

Rose-Hulman Institute of Technology

Final Exam

Email me by Tuesday, Feb. 16, to sign up.

- Monday, Feb. 22, 8am
- **Optional**
  - If you don't take the exam, we'll use your exam 1 grade as your final exam grade
  - Sign-up for exam during 10th week
  - If you sign-up, you have to take the exam
  - Taking the exam can lower your grade

Plan for Today

- Short survey on projects
- State Pattern
- Command Pattern
- Design Studio—Concurrent Poker Player

Please bring laptops tomorrow for course evaluations.

http://flic.kr/p/46Hca

Checkers

Handling Turn Taking and Undo

Turn Taking in Checkers

- **Simple move**: slide a piece diagonally to adjacent, open square
- **Jump move**: move a piece diagonally over an adjacent opponents piece landing in an open square
- **Multiple jumps**: *must* jump again if another jump is available after landing

http://en.wikipedia.org/wiki/English_draughts

It's Good to be King

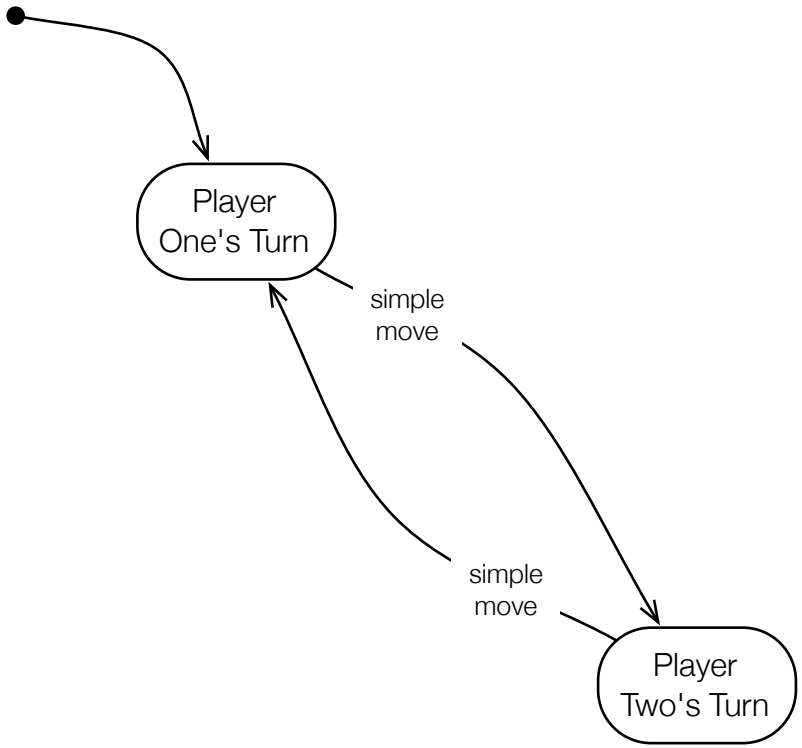
- A piece reaching the far side of the board is **kinged**
- Only **kings** may move "backward" (toward the player)
- A player's turn ends when a piece is kinged (i.e., can't jump into king row, become a king, and jump back out)

http://en.wikipedia.org/wiki/English_draughts

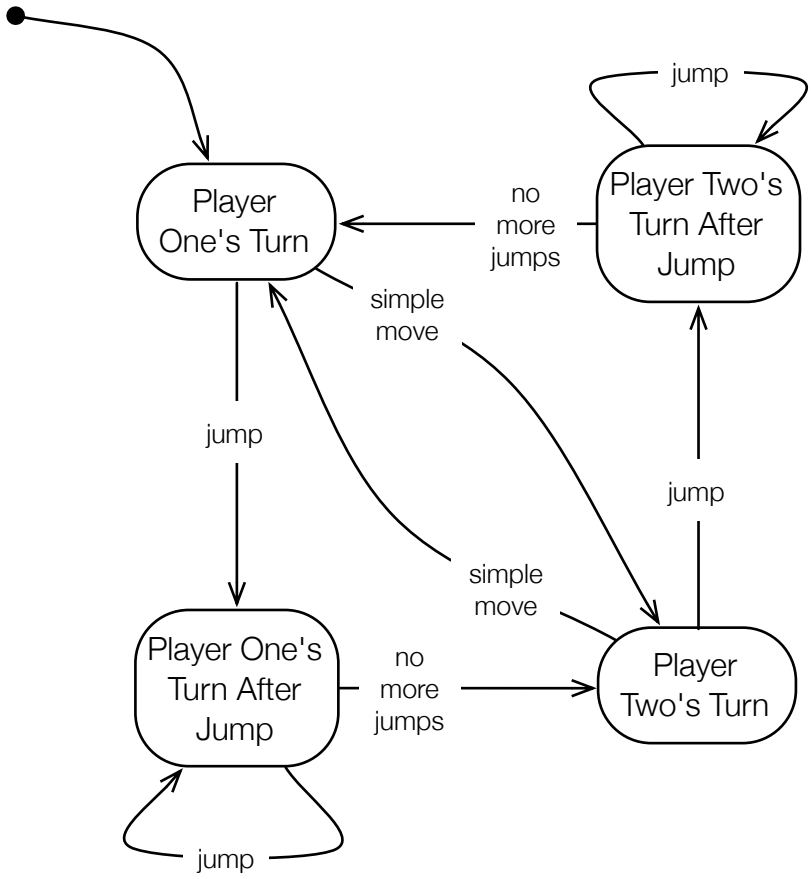
State Machines

- Good way to represent the behavior of a system when future actions depend on previous actions

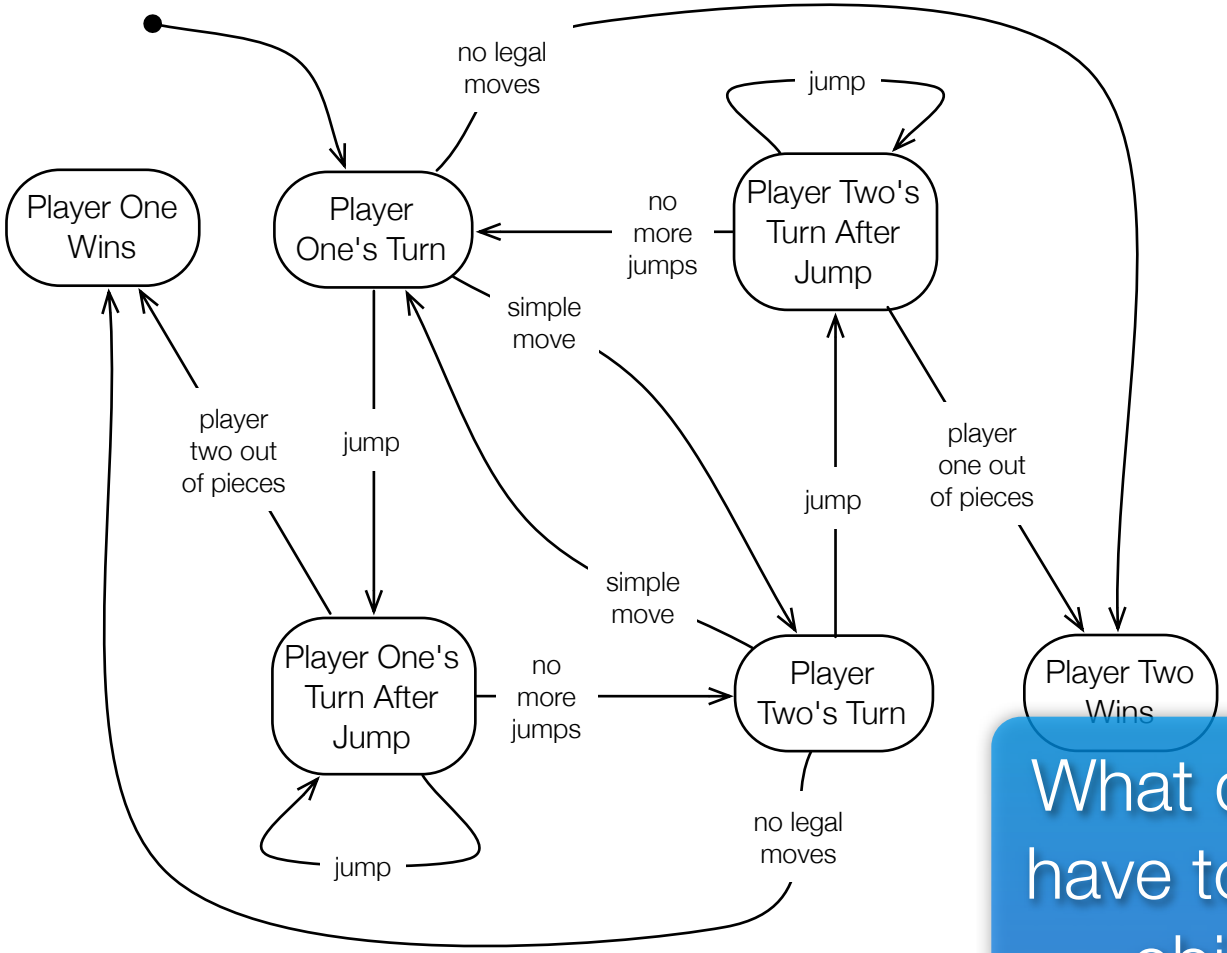
Basic Turn Taking



Jumping

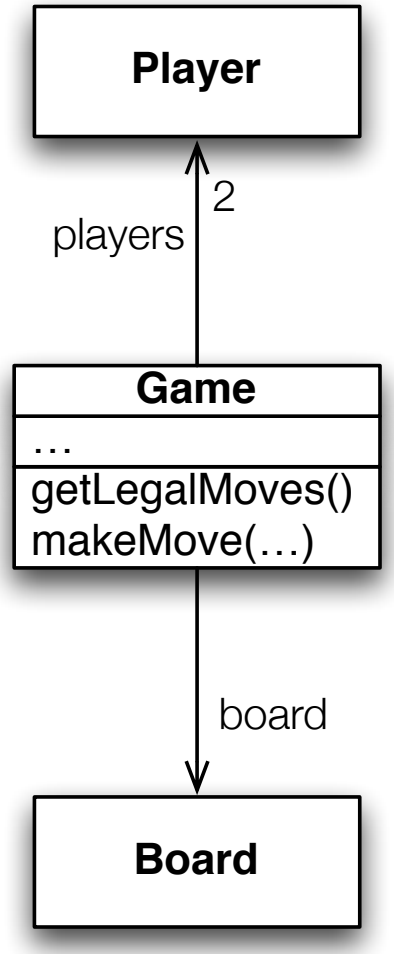

jump
Player
One's Turn
no
more
jumps
Player Two's
Turn After
Jump
simple
move
jump
jump
simple
move
Player One's
Turn After
Jump
no
more
jumps
Player
Two's Turn
jump

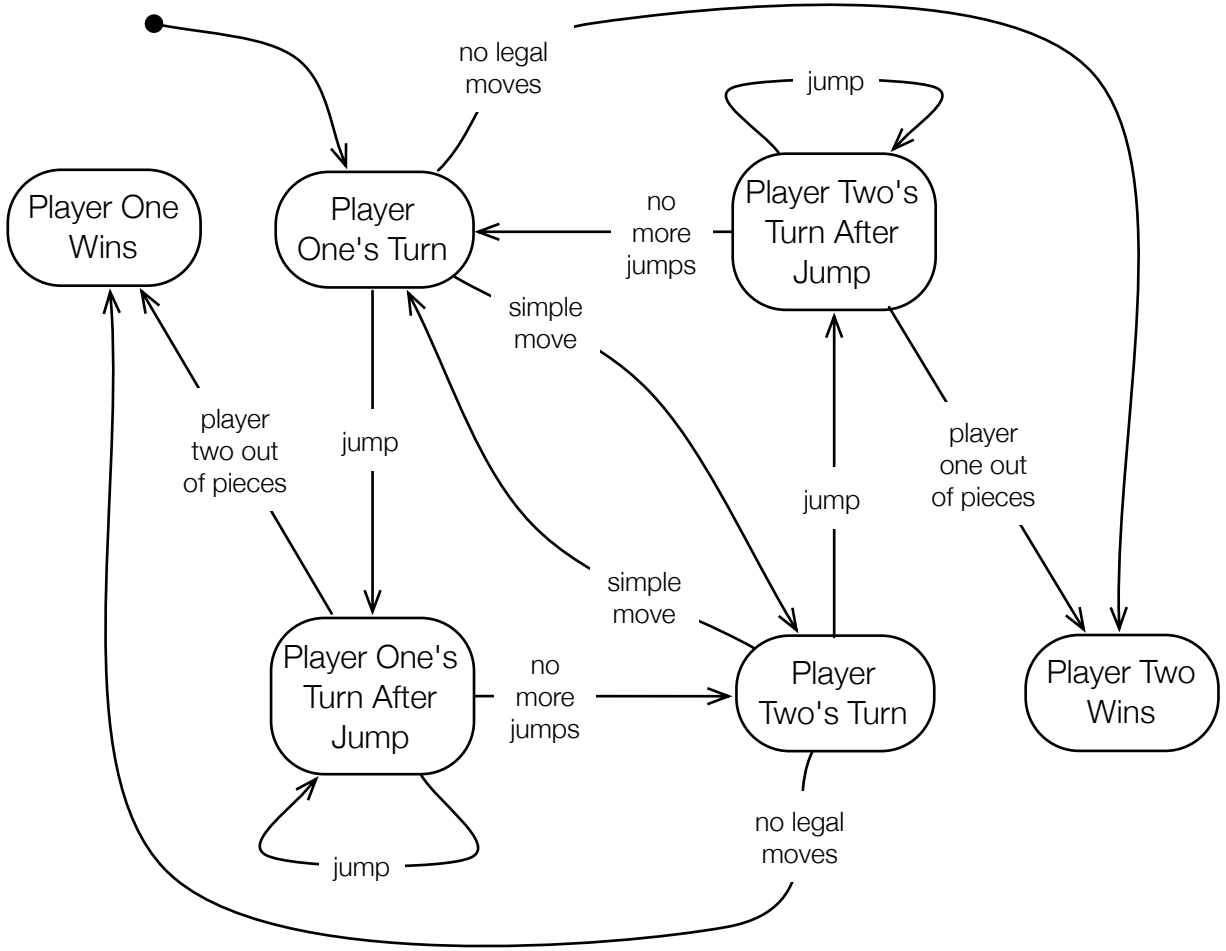
Winners and Losers



What does this have to do with objects?

Winners and Losers



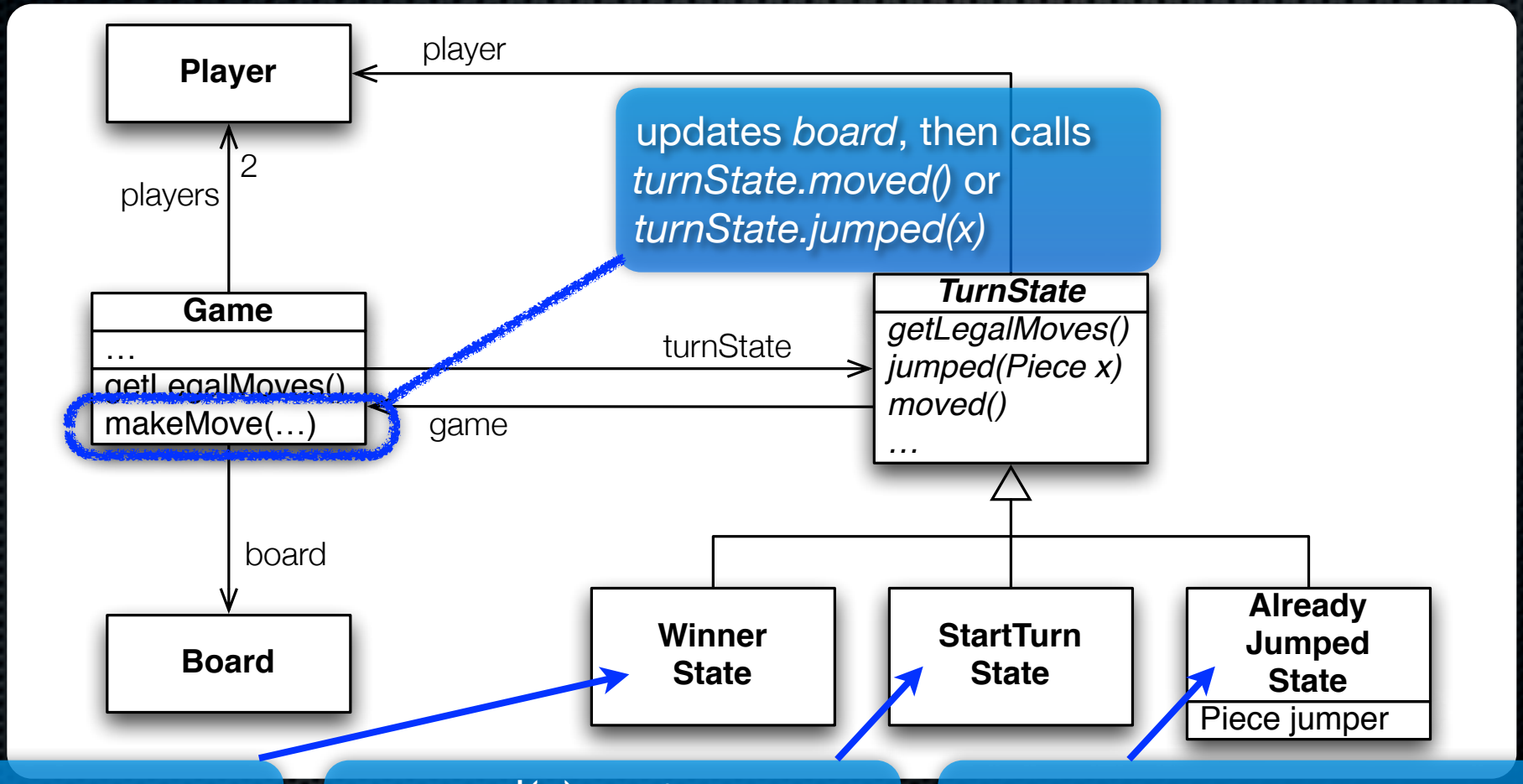


State Pattern

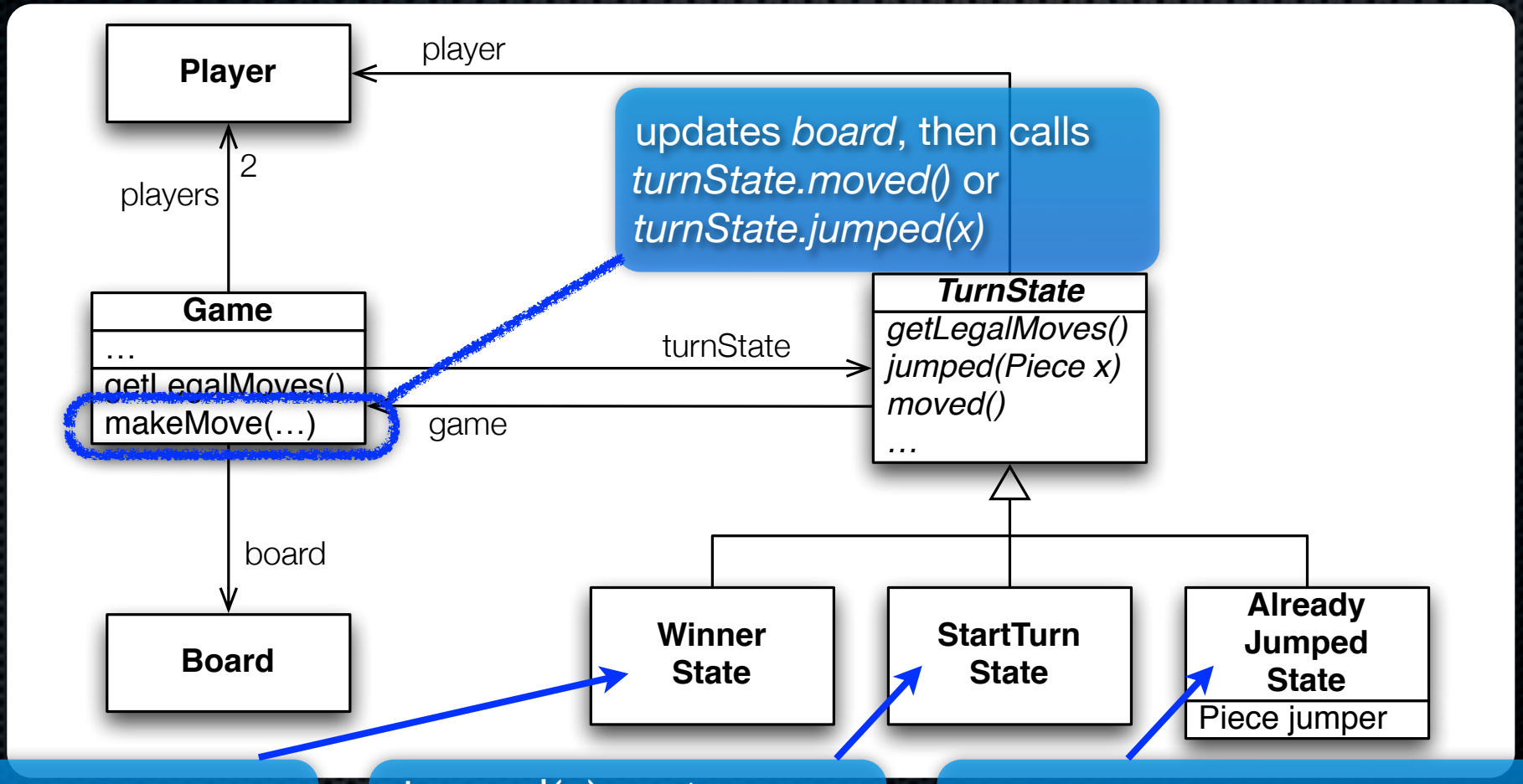
- **Problem**: When the behavior of an object, *obj*, changes depending on its state, how can we avoid complicated conditional statements?
- **Solution**: Create **state** classes implementing a common interface. Delegate state-dependent methods from *obj* to the current **state** object.
- Example…

Q1,2

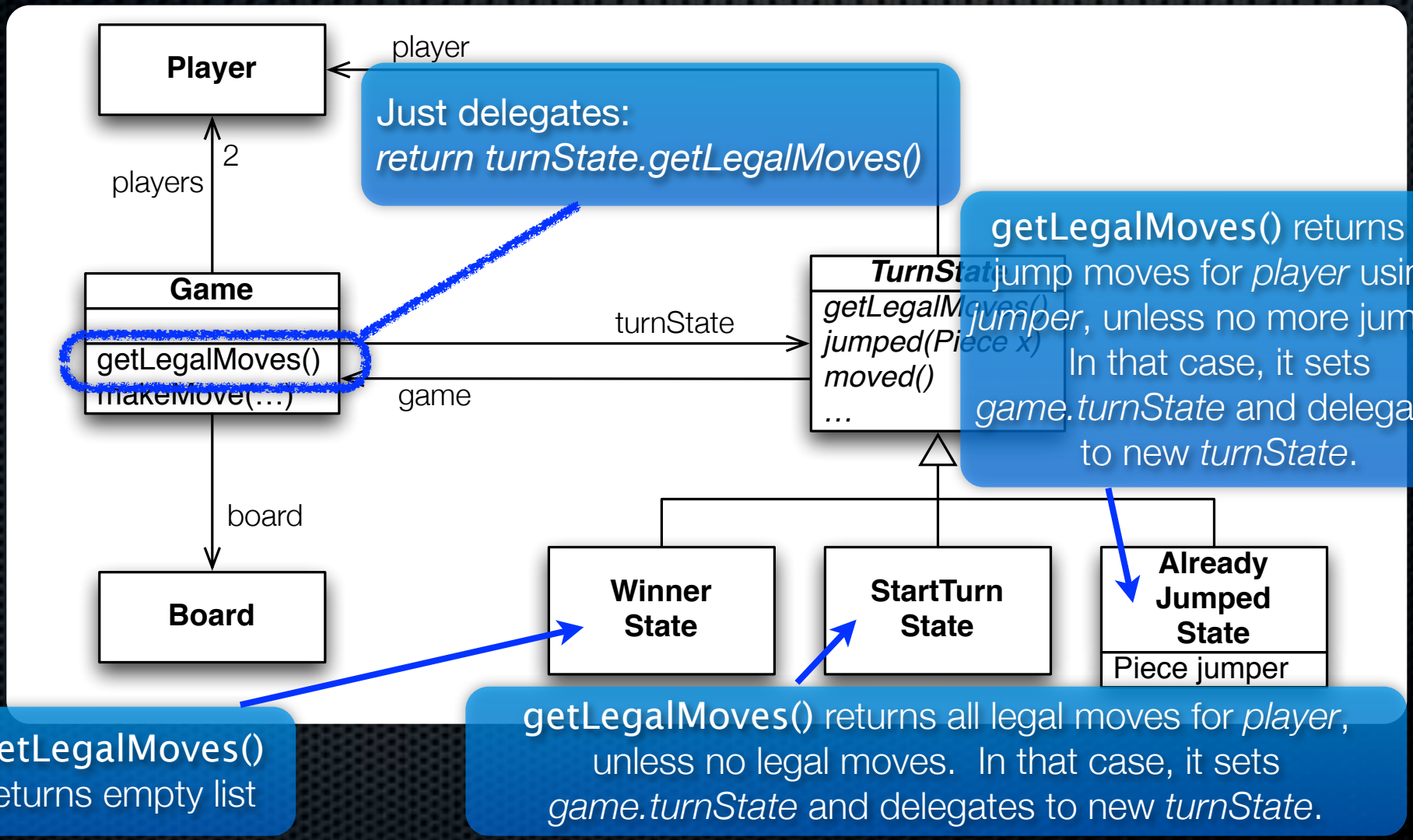
Handling Simple Moves



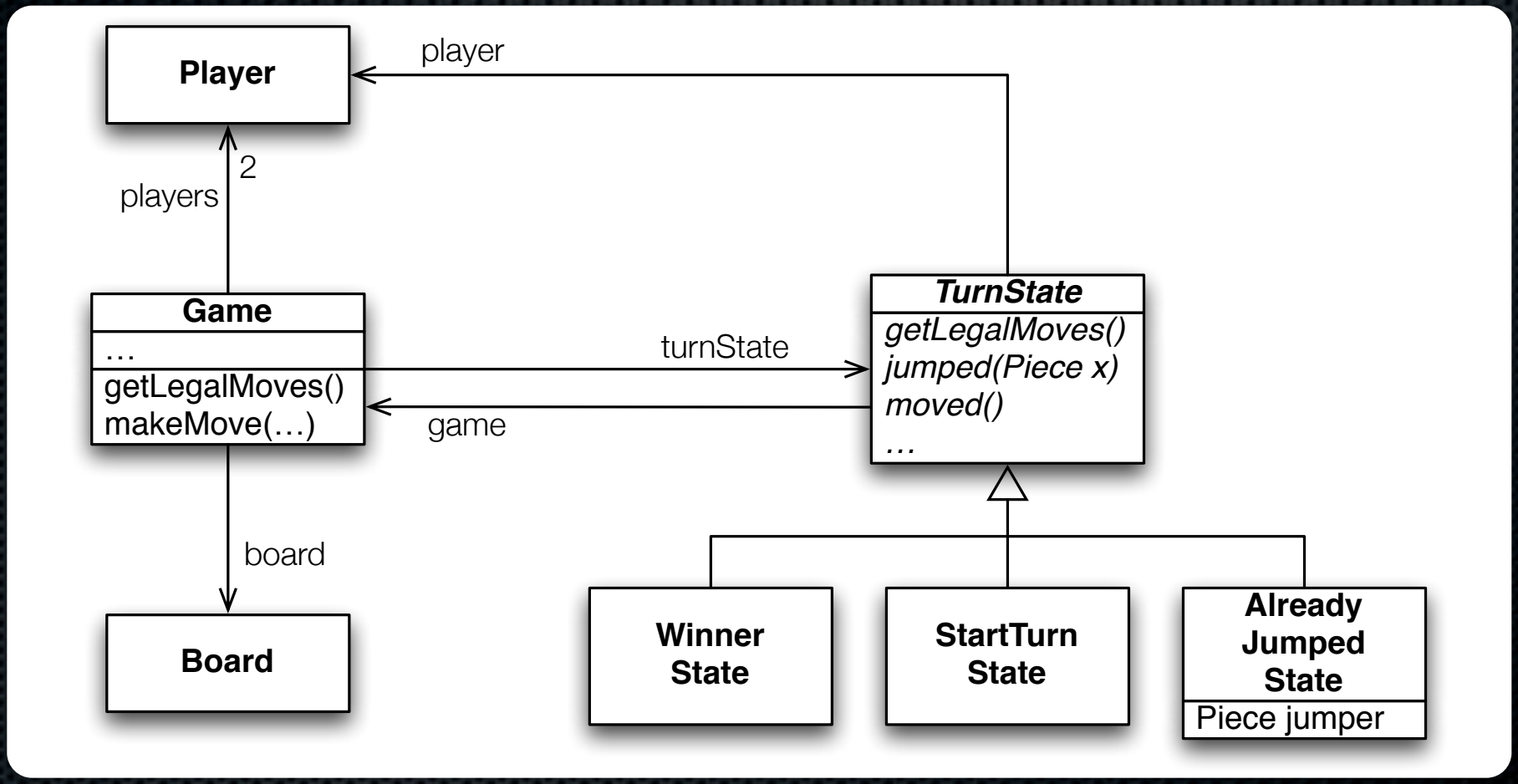
Handling Jump Moves



Getting Legal Moves

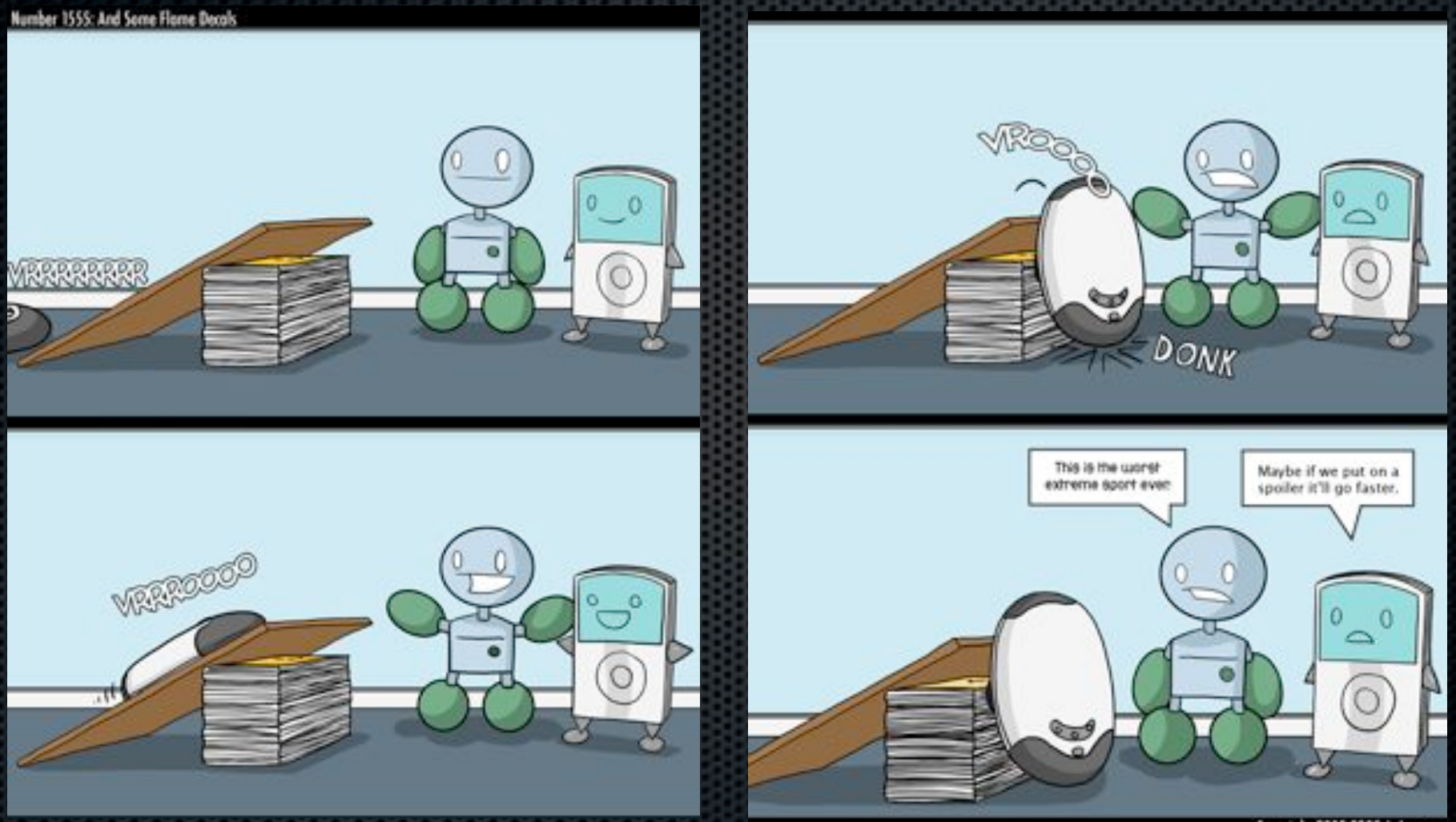


Applying the State Pattern



Q3

Cartoon of the Day



Used by permission. http://www.questionablecontent.net/view.php?comic=1555

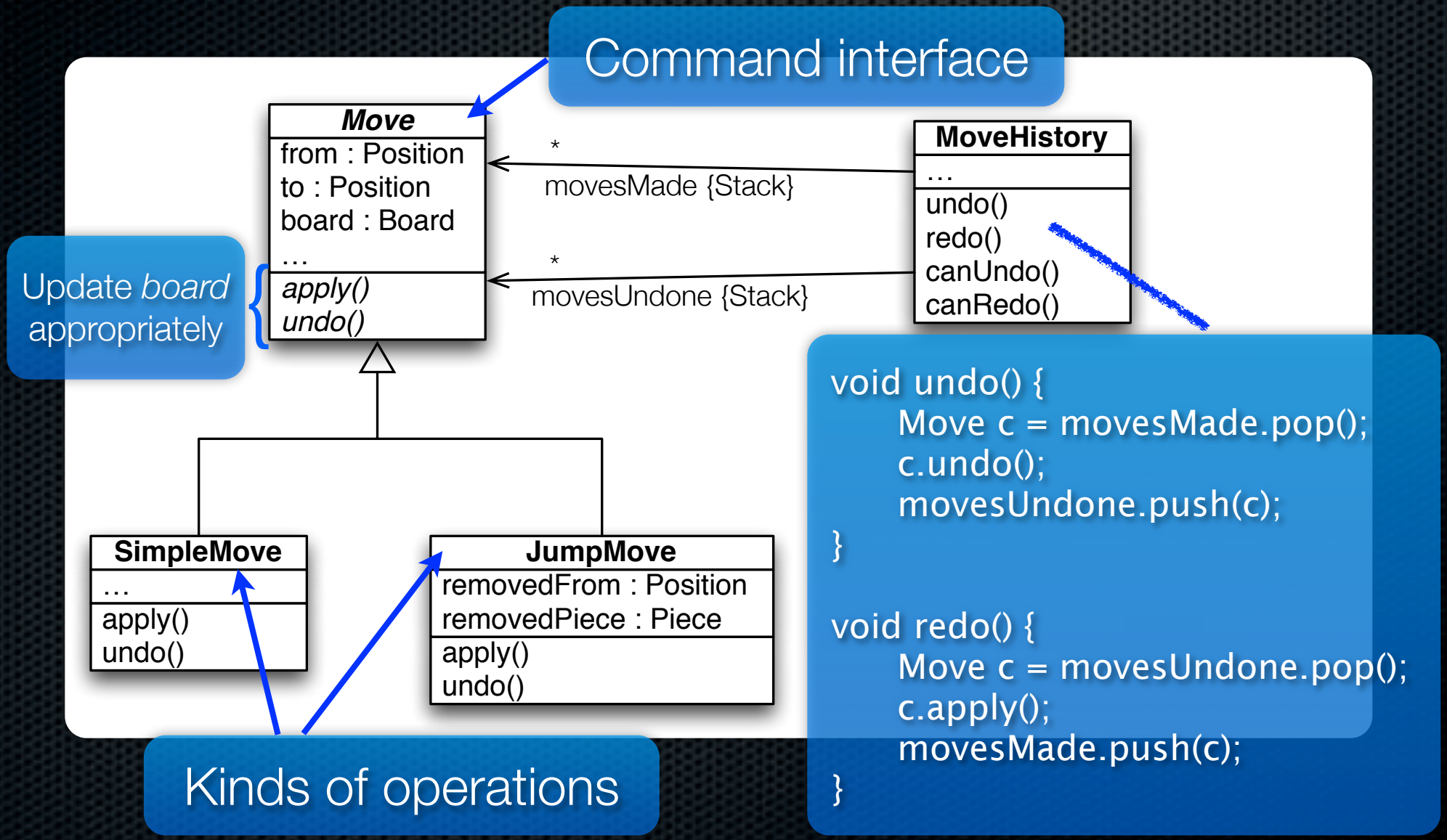
Suppose we want to be able to undo moves

Command Pattern

- **Problem**: When we need to record operations so we can undo them, or execute them later, what should we do?
- **Solution**: Define a **Command** interface that represents all possible operations. Create subclasses of it for each kind of operation and instances for each actual operation.
- Example…

Q4,5

Adding Undo to Checkers



Uses for the Command Pattern

- Undo/redo
- Prioritizing and Queueing operations
- Composing multi-part operations
- Progress bars
- Macro recording

Design Studio: Concurrent Poker Player

Team describes problem and perhaps current solution (if any)	~5 min.
Class thinks about questions, alternative approaches. **Q7**	~3 min.
On-board design	~12 min.