CardMurray
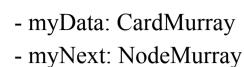
- myValue: int
- mySuit: char

+ CardMurray(newValue: int, newSuit: char)
+ CardMurray()
+ setValue(newValue: int): void
+setSuit(newSuit: char): void
+getValue(): int
+getSuit(): char

Occupies →

NodeMurray

- myData: CardMurray
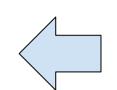- myNext: NodeMurray

+ NodeMurray(newData: CardMurray)
+ NodeMurray()
+ setData(newData: CardMurray): void
+ setNext(newNext: NodeMurray): void
+getData(): CardMurray
+getNext(): NodeMurray

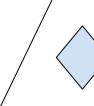**Contains** ←

StackMurray

-myHead: NodeMurray

+ Stack()
+ setHead(newHead: NodeMurray): void
+ getHead(): NodeMurray
+push(value: int): Boolean
+ pop(): int
+ isEmpty(): Boolean
+isFull(): Boolean

1   *

1...52   1

◇ Composition (Open Diamond): BattleDemoMurray has a StackMurray

BattleDemoMurray

+deal(playerStack1: StackMurray, playerStack2: StackMurray): int
+play(playerStack1: StackMurray, playerStack2: StackMurray,
    discard1: StackMurray, discard2: StackMurray): void
+compare(poppedValues: StackMurray, discard1: StackMurray, discard2: StackMurray): void
+winPlayer1(card1: CardMurray, card2: CardMurray, discardStack1: StackMurray): void
+winPlayer2(card1: CardMurray, card2: CardMurray, discardStack2: StackMurray): void
+ tie(card1: StackMurray, card2: StackMurray, discardStack1: StackMurray,
    discardStack2: StackMurray): void
+copyPlayer1(player1Stack: StackMurray, discard: StackMurray): StackMurray
+copyPlayer2(player2Stack: StackMurray, discard: StackMurray): StackMurray
+countCards(cards: StackMurray): int
+printResults(numCards: int, plays: int, player1play: StackMurray, player2play: StackMurray,
      player1discard: StackMurray, player2discard: StackMurray): void