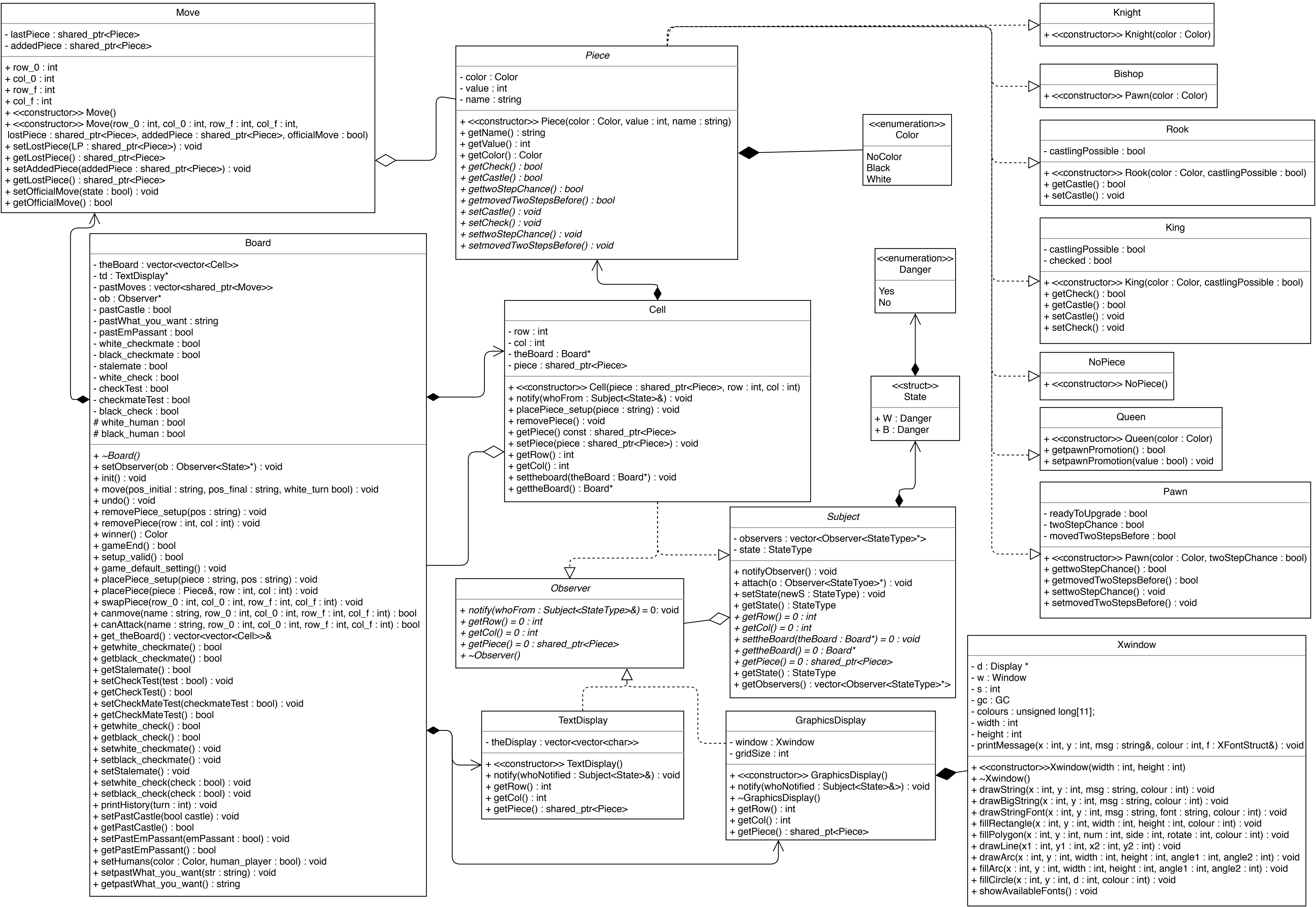
Winter 2019 CS246 A5 Chess UML Diagram



```
Move
- lastPiece : shared_ptr<Piece>
- addedPiece : shared_ptr<Piece>
+ row_0 : int
+ col_0 : int
+ row_f : int
+ col_f : int
+ <<constructor>> Move()
+ <<constructor>> Move(row_0 : int, col_0 : int, row_f : int, col_f : int,
 lostPiece : shared_ptr<Piece>, addedPiece : shared_ptr<Piece>, officialMove : bool)
+ setLostPiece(LP : shared_ptr<Piece>) : void
+ getLostPiece() : shared_ptr<Piece>
+ setAddedPiece(addedPiece : shared_ptr<Piece>) : void
+ getLostPiece() : shared_ptr<Piece>
+ setOfficialMove(state : bool) : void
+ getOfficialMove() : bool
```

```
Piece
- color : Color
- value : int
- name : string
+ <<constructor>> Piece(color : Color, value : int, name : string)
+ getName() : string
+ getValue() : int
+ getColor() : Color
+ getCheck() : bool
+ getCastle() : bool
+ gettwoStepChance() : bool
+ getmovedTwoStepsBefore() : bool
+ setCastle() : void
+ setCheck() : void
+ settwoStepChance() : void
+ setmovedTwoStepsBefore() : void
```

```
<<enumeration>>
Color
NoColor
Black
White
```

```
Knight
+ <<constructor>> Knight(color : Color)
```

```
Bishop
+ <<constructor>> Pawn(color : Color)
```

```
Rook
- castlingPossible : bool
+ <<constructor>> Rook(color : Color, castlingPossible : bool)
+ getCastle() : bool
+ setCastle() : void
```

```
King
- castlingPossible : bool
- checked : bool
+ <<constructor>> King(color : Color, castlingPossible : bool)
+ getCheck() : bool
+ getCastle() : bool
+ setCastle() : void
+ setCheck() : void
```

```
NoPiece
+ <<constructor>> NoPiece()
```

```
Queen
+ <<constructor>> Queen(color : Color)
+ getpawnPromotion() : bool
+ setpawnPromotion(value : bool) : void
```

```
Pawn
- readyToUpgrade : bool
- twoStepChance : bool
- movedTwoStepsBefore : bool
+ <<constructor>> Pawn(color : Color, twoStepChance : bool)
+ gettwoStepChance() : bool
+ getmovedTwoStepsBefore() : bool
+ settwoStepChance() : void
+ setmovedTwoStepsBefore() : void
```

```
<<enumeration>>
Danger
Yes
No
```

```
<<struct>>
State
+ W : Danger
+ B : Danger
```

```
Board
- theBoard : vector<vector<Cell>>
- td : TextDisplay*
- pastMoves : vector<shared_ptr<Move>>
- ob : Observer*
- pastCastle : bool
- pastWhat_you_want : string
- pastEmPassant : bool
- white_checkmate : bool
- black_checkmate : bool
- stalemate : bool
- white_check : bool
- checkTest : bool
- checkmateTest : bool
- black_check : bool
# white_human : bool
# black_human : bool
+ ~Board()
+ setObserver(ob : Observer<State>*) : void
+ init() : void
+ move(pos_initial : string, pos_final : string, white_turn bool) : void
+ undo() : void
+ removePiece_setup(pos : string) : void
+ removePiece(row : int, col : int) : void
+ winner() : Color
+ gameEnd() : bool
+ setup_valid() : bool
+ game_default_setting() : void
+ placePiece_setup(piece : string, pos : string) : void
+ placePiece(piece : Piece&, row : int, col : int) : void
+ swapPiece(row_0 : int, col_0 : int, row_f : int, col_f : int) : void
+ canmove(name : string, row_0 : int, col_0 : int, row_f : int, col_f : int) : bool
+ canAttack(name : string, row_0 : int, col_0 : int, row_f : int, col_f : int) : bool
+ get_theBoard() : vector<vector<Cell>>&
+ getwhite_checkmate() : bool
+ getblack_checkmate() : bool
+ getStalemate() : bool
+ setCheckTest(test : bool) : void
+ getCheckTest() : bool
+ setCheckMateTest(checkmateTest : bool) : void
+ getCheckMateTest() : bool
+ getwhite_check() : bool
+ getblack_check() : bool
+ setwhite_checkmate() : void
+ setblack_checkmate() : void
+ setStalemate() : void
+ setwhite_check(check : bool) : void
+ setblack_check(check : bool) : void
+ printHistory(turn : int) : void
+ setPastCastle(bool castle) : void
+ getPastCastle() : bool
+ setPastEmPassant(emPassant : bool) : void
+ getPastEmPassant() : bool
+ setHumans(color : Color, human_player : bool) : void
+ setpastWhat_you_want(str : string) : void
+ getpastWhat_you_want() : string
```

```
Cell
- row : int
- col : int
- theBoard : Board*
- piece : shared_ptr<Piece>
+ <<constructor>> Cell(piece : shared_ptr<Piece>, row : int, col : int)
+ notify(whoFrom : Subject<State>&) : void
+ placePiece_setup(piece : string) : void
+ removePiece() : void
+ getPiece() const : shared_ptr<Piece>
+ setPiece(piece : shared_ptr<Piece>) : void
+ getRow() : int
+ getCol() : int
+ settheboard(theBoard : Board*) : void
+ gettheBoard() : Board*
```

```
Observer
+ notify(whoFrom : Subject<StateType>&) = 0: void
+ getRow() = 0 : int
+ getCol() = 0 : int
+ getPiece() = 0 : shared_ptr<Piece>
+ ~Observer()
```

```
Subject
- observers : vector<Observer<StateType>*>
- state : StateType
+ notifyObserver() : void
+ attach(o : Observer<StateTyoe>*) : void
+ setState(newS : StateType) : void
+ getState() : StateType
+ getRow() = 0 : int
+ getCol() = 0 : int
+ settheBoard(theBoard : Board*) = 0 : void
+ gettheBoard() = 0 : Board*
+ getPiece() = 0 : shared_ptr<Piece>
+ getState() : StateType
+ getObservers() : vector<Observer<StateType>*>
```

```
TextDisplay
- theDisplay : vector<vector<char>>
+ <<constructor>> TextDisplay()
+ notify(whoNotified : Subject<State>&) : void
+ getRow() : int
+ getCol() : int
+ getPiece() : shared_ptr<Piece>
```

```
GraphicsDisplay
- window : Xwindow
- gridSize : int
+ <<constructor>> GraphicsDisplay()
+ notify(whoNotified : Subject<State>&>) : void
+ ~GraphicsDisplay()
+ getRow() : int
+ getCol() : int
+ getPiece() : shared_pt<Piece>
```

```
Xwindow
- d : Display *
- w : Window
- s : int
- gc : GC
- colours : unsigned long[11];
- width : int
- height : int
- printMessage(x : int, y : int, msg : string&, colour : int, f : XFontStruct&) : void
+ <<constructor>>Xwindow(width : int, height : int)
+ ~Xwindow()
+ drawString(x : int, y : int, msg : string, colour : int) : void
+ drawBigString(x : int, y : int, msg : string, colour : int) : void
+ drawStringFont(x : int, y : int, msg : string, font : string, colour : int) : void
+ fillRectangle(x : int, y : int, width : int, height : int, colour : int) : void
+ fillPolygon(x : int, y : int, num : int, side : int, rotate : int, colour : int) : void
+ drawLine(x1 : int, y1 : int, x2 : int, y2 : int) : void
+ drawArc(x : int, y : int, width : int, height : int, angle1 : int, angle2 : int) : void
+ fillArc(x : int, y : int, width : int, height : int, angle1 : int, angle2 : int) : void
+ fillCircle(x : int, y : int, d : int, colour : int) : void
+ showAvailableFonts() : void
```