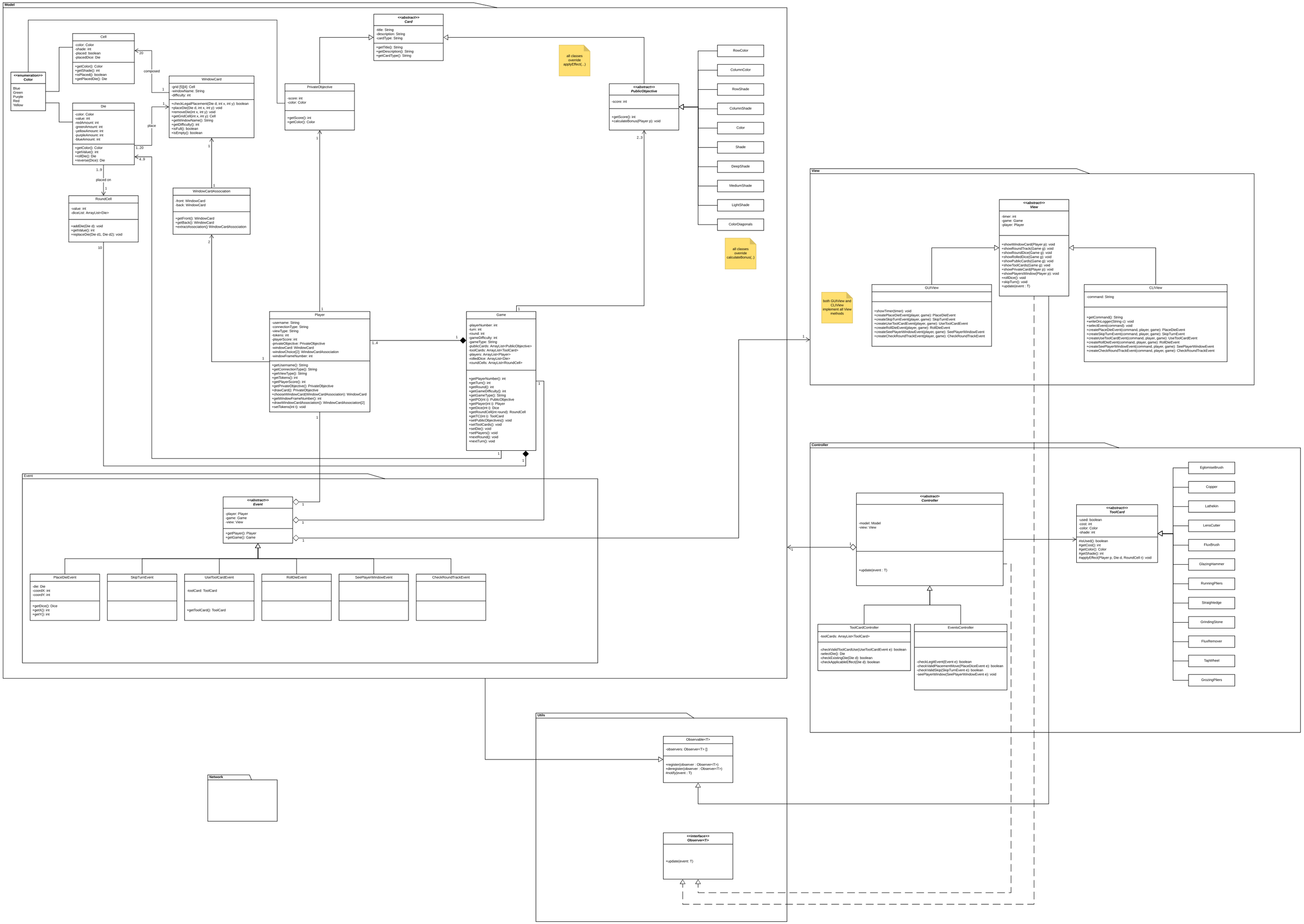
Model
<<abstract>>
Card
-title: String
-description: String
-cardType: String
+getTitle(): String
+getDescription(): String
+getCardType(): String
all classes override applyEffect(...)
Cell
-color: Color
-shade: int
-placed: boolean
-placedDice: Die
+getColor(): Color
+getShade(): int
+isPlaced(): boolean
+getPlacedDie(): Die
20
composed
1
<<enumeration>>
Color
Blue
Green
Purple
Red
Yellow
Die
-color: Color
-value: int
-redAmount: int
-greenAmount: int
-yellowAmount: int
-purpleAmount: int
-blueAmount: int
+getColor(): Color
+getValue(): int
+rollDie(): Die
+reverse(Dice): Die
1..20
place
4..9
1..9
placed on
1
WindowCard
-grid [5][4]: Cell
-windowName: String
-difficulty: int
+checkLegalPlacement(Die d, int x, int y): boolean
+placeDie(Die d, int x, int y): void
+removeDie(int x, int y): void
+getGridCell(int x, int y): Cell
+getWindowName(): String
+getDifficulty(): int
+isFull(): boolean
+isEmpty(): boolean
1
1
PrivateObjective
-score: int
-color: Color
+getScore(): int
+getColor(): Color
1
<<abstract>>
PublicObjective
-score: int
+getScore(): int
+calculateBonus(Player p): void
2..3
RowColor
ColumnColor
RowShade
ColumnShade
Color
Shade
DeepShade
MediumShade
LightShade
ColorDiagonals
all classes override calculateBonus(..)
RoundCell
-value: int
-diceList: ArrayList<Die>
+addDie(Die d): void
+getValue(): int
+replaceDie(Die d1, Die d2): void
10
WindowCardAssociation
-front: WindowCard
-back: WindowCard
+getFront(): WindowCard
+getBack(): WindowCard
+extractAssociation(): WindowCardAssociation
2
Player
-username: String
-connectionType: String
-viewType: String
-tokens: int
-playerScore: int
-privateObjective: PrivateObjective
-windowCard: WindowCard
-windowChoice[2]: WindowCardAssociation
-windowFrameNumber: int
+getUsername(): String
+getConnectionType(): String
+getViewType(): String
+getTokens(): int
+getPlayerScore(): int
+getPrivateObjective(): PrivateObjective
+drawCard(): PrivateObjective
+chooseWindowCard(WindowCardAssociation): WindowCard
+getWindowFrameNumber(): int
+drawWindowCardAssociation(): WindowCardAssociation[2]
+setTokens(int t): void
1
1..4
1
1
Game
-playerNumber: int
-turn: int
-round: int
-gameDifficulty: int
-gameType: String
-publicCards: ArrayList<PublicObjective>
-toolCards: ArrayList<ToolCard>
-players: ArrayList<Player>
-rolledDice: ArrayList<Die>
-roundCells: ArrayList<RoundCell>
+getPlayerNumber(): int
+getTurn(): int
+getRound(): int
+getGameDifficulty(): int
+getGameType(): String
+getPO(int i): PublicObjective
+getPlayer(int i): Player
+getDice(int i): Dice
+getRoundCell(int round): RoundCell
+getTC(int i): ToolCard
+setPublicObjectives(): void
+setToolCards(): void
+setDie(): void
+setPlayers(): void
+nextRound(): void
+nextTurn(): void
1
1
1
1
Event
<<abstract>>
Event
-player: Player
-game: Game
-view: View
+getPlayer(): Player
+getGame(): Game
1
1
1
PlaceDieEvent
-die: Die
-coordX: int
-coordY: int
+getDice(): Dice
+getX(): int
+getY(): int
SkipTurnEvent
UseToolCardEvent
-toolCard: ToolCard
+getToolCard(): ToolCard
RollDieEvent
SeePlayerWindowEvent
CheckRoundTrackEvent
View
<<abstract>>
View
-timer: int
-game: Game
-player: Player
+showWindowCard(Player p): void
+showRoundTrack(Game g): void
+showRoundDice(Game g): void
+showRolledDice(Game g): void
+showPublicCards(Game g): void
+showToolCards(Game g): void
+showPrivateCard(Player p): void
+showPlayersWindow(Player p): void
+rollDice(): void
+skipTurn(): void
+update(event : T)
both: GUIView and CLIView implement all View methods
GUIView
+showTimer(timer): void
+createPlaceDieEvent(player, game): PlaceDieEvent
+createSkipTurnEvent(player, game): SkipTurnEvent
+createUseToolCardEvent(player, game): UseToolCardEvent
+createRollDieEvent(player, game): RollDieEvent
+createSeePlayerWindowEvent(player, game): SeePlayerWindowEvent
+createCheckRoundTrackEvent(player, game): CheckRoundTrackEvent
CLIView
-command: String
+getCommand(): String
+writeOnLogger(String c): void
+selectEvent(command): void
+createPlaceDieEvent(command, player, game): PlaceDieEvent
+createSkipTurnEvent(command, player, game): SkipTurnEvent
+createUseToolCardEvent(command, player, game): UseToolCardEvent
+createRollDieEvent(command, player, game): RollDieEvent
+createSeePlayerWindowEvent(command, player, game): SeePlayerWindowEvent
+createCheckRoundTrackEvent(command, player, game): CheckRoundTrackEvent
1
Controller
<<abstract>>
Controller
-model: Model
-view: View
+update(event : T)
1
1
ToolCardController
-toolCards: ArrayList<ToolCard>
-checkValidToolCardUse(UseToolCardEvent e): boolean
-selectDie(): Die
-checkExistingDie(Die d): boolean
-checkApplicableEffect(Die d): boolean
EventsController
-checkLegitEvent(Event e): boolean
-checkValidPlacementMove(PlaceDiceEvent e): boolean
-checkValidSkipTurnEvent e): boolean
-seePlayerWindow(SeePlayerWindowEvent e): void
<<abstract>>
ToolCard
-used: boolean
-cost: int
-color: Color
-shade: int
#isUsed(): boolean
#getCost(): int
#getColor(): Color
#getShade(): int
#applyEffect(Player p, Die d, RoundCell r): void
EglomiseBrush
Copper
Lathekin
LensCutter
FluxBrush
GlazingHammer
RunningPliers
Straightedge
GrindingStone
FluxRemover
TapWheel
GrozingPliers
Utils
Observable<T>
-observers: Observer<T> []
+register(observer : Observer<T>)
+deregister(observer : Observer<T>)
#notify(event : T)
<<interface>>
Observer<T>
+update(event: T)
Network