Roger Zheng | October 4, 2019 Carcassonne -gameOver:bool: //Generals +main():void; +initate():void; -getTile(posX:int,posY:int):Tile; //Interactions +placeTile(p:Player,s:Tile,posX:int,posY:int):bool; +rotateTileCW(s:Tile,time:int):bool; +rotateTileCCW(s:Tile,time:int):bool; +placeMeeple(p:Player,e:Edge,s:Tile):bool; +requestTile():Tile; //with auto discard request file(). file; //wint auto disadri -removeMeeple(p:Player,e:Edge,s:Tile):bool; -findAllOpenPositions():List<-int[2]{posX,posY}>; -checkTilePlaceable(t:Tile):bool; //for all empty positions -checkTileValid(t:Tile):bool; //with adjacent Tiles -checkTileAtValid(posX:int,posY:int):bool; Updated By 2..5 Player -checkMeeplePlaceable(t:Tile):bool; //within this Tile -checkMeepleValid(t:Tile):bool; //within this Feature -name:String; Reports To -score:int; //Rules -placedMeeples:List<Meeples>; ScoreController +checkEnd():bool; +getWinner():Player; +getName():string; +getScore():int; +scoreEndGamePlayer(p:Player,b:Board):void; -scoreMeeple(p:Player,e:Edge,s:Tile); +addScore(sc:int):int;//returns old score +scoreUpdate(p:Player,b:Board,t:Tile):void; -findAllRelatedTiles(t:Tile):List<Tile> +setScore(sc:int):int;//returns old score -checkCompleteFeature(p:Player):int; +getPlacedMeeples():List<Meeple>; -updateScore(p:Player):void; +getMeepleAtTile(t:Tile):Meeple; //null if non exist -calcEndgameScore():void; +getFeaturedMeeples(f:Feature):List<Meeple>; +removeMeeple(m:Meeple); **Board** Deck -Map<Integer.Tile>: -List<Integer> placeablePosition; //Tiles stored in ArrayList +updateBoard():void; +getTop():Tile; +getTile(posX:int,posY:int):Tile; +getRemaining():int; +isAdjTiles(t1:Tile,t2:Tile); -initiateDeck():void; +getAdjTiles(t:Tile):List<Tile>; +findTile(t:Tile):int[3]{exist, posX, posY}; +adjTileMatch(t1:Tile,t2:Tile):bool; -hashPos(posX:int,posY:int):int; -featureMatch(f1:Feature,f2:Feature):bool; 0..72 Tile Meeple -tileType:TileType; -t:Tile; -e:Edge; -isOpen:bool; //false if in deck -vaussO--isFinal:bool; //final lock +occupiedFeature():Feature; -northFeature:Feature: +occupiedTile():Tile; -southFeature:Feature; -eastFeature:Feature; -westFeature:Feature; -centerFeature:Feature; -northwestCornerFeature:CornerFeature; TileParser -northeastCornerFeature:CornerFeature; -southwestCornerFeature:CornerFeature; -jsonParseFile:File; -southeastCornerFeature:CornerFeature; -contentRotation: Edge; 0..72 -posX:int; +tileEdgeFeature(tt:TileType, e:Edge):Feature; -posY:int; +tileCornerFeatures(tt:TileType, c:Corner):CornerFeature; <enum> Edge {North; South; East; West; Center}; +checkOpen():bool; +getMeeple():List<Meeple>; +getMeeple(e:Edge):Meeple;//or null +getEdgeFeature(e:Edge):Feature; <enum> Corner +getCornerFeature(c:Corner):CornerFeature; <enum> Feature +getTileType():TileType; {NorthEast; SouthEast; NorthWest; +getRotation():Edge;//current edge of original north SouthWest); {Road_Connect; Field_Connect; +setTile(t:TileType):void;//can add final lock <enum> TileType +rotatePiece(times:int);//clockwise, update; City_Connect; Monastery; City; City_Guarded; Field;}; +open():void; <enum> CornerFeature +finalize():void; {Tile_A; Tile_B; ...; Tile_X}; +getPosX():int;//throws exception if tile not closed; {City_Crossing; City_Corner}; //Only features that take >=50% can be +getPosY():int;//throws exception if tile not closed; assigned to the center;