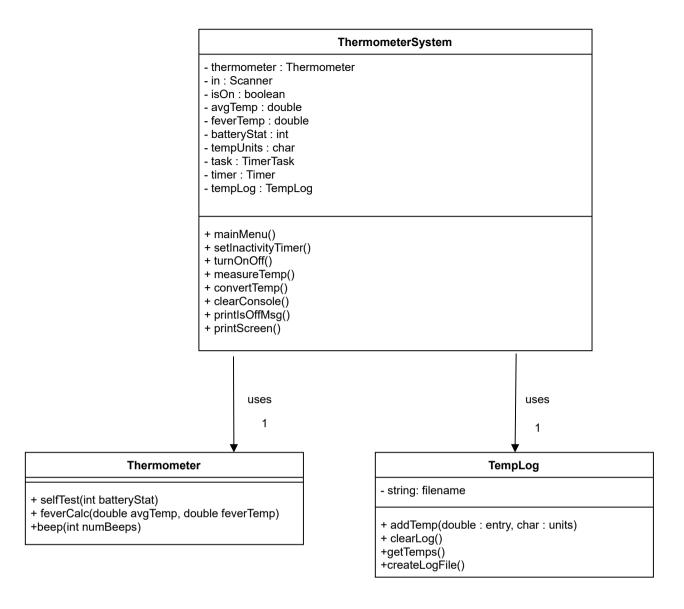
Thermometer Complete UML Class Diagram



Sudoku Analyzer UML Class Diagram, page 2 (see notes below diagram) version 1 (partial)

Puzzle Unit + name : String //Ri, Cj, Bk i,j,k = 1..9 - isComplete : Boolean //fully solved + name : String //puzzle name or null - contents : Square[9] - board : Square[0..8, 0..8] // but called 1 to 9 - fromFile : //some File type - indexNumber : INumber //in allCols, allRows, allBoxes - allUnits: Unit[27] //all rows, columns, boxes; needed?? has - unitNumber: int // 0..26, may not need - allRows : Unit[9] - allColumns : Unit[9] - allBoxes : Unit[9] - solvedNumbers: Set<ANumber> // numbers 1..9 completed + possibleSquares(n:ANumber) : List<Squares> //null if n - isConsistent : Boolean //full puzzle; all other levels calculate on access already in Unit + isComplete: Boolean // all squares filled and consistent + level : String //null if unknown + numInitial: int //given initial values + numFilled : int //numInitial + squaresSolved + numMoves : int //same as squaresSolved?? - timeUsed: //some Time class (solution time); possibly multiple runs - solutionHistory : List<Square> //in order solved; allows undo - pencilHistory: List<CandidateChange> //in order erased Extends Extends + <<constructor>> (originalContent:String[81], solution:String[81], currentContent:String[81]) //solution and currentContent may be all 0s + pencillnAll(): Boolean //false if candidates already present in puzzle + getSquare(row:ANumber, Col:ANumber) : Square Row Column + field: type + field: type + field: type has see below see below see below + <<constructor>>(num:ANumber) + associateSquare(s:Square) : Boolean //false if > 9 attempted 81 Square - content : SquareNumber //0 -> empty + isGiven : final Boolean //initial starting number - solution : SquareNumber //used if solution also given in input file - candidates : Set<ANumber> + name : String //RiCj i,j = 1..9 Note on Box number calculation within Square constructor - mvRow : Unit Given r and c as INumbers (0..8), square name is 'r'(r+1)'c'(c+1) - myColumn : Unit boxNumber = (r/3)*3 + (c/3) + 1 //int arithmetic - myBox : Unit + <<constructor>>(row:ANumber, col:ANumber, r: Row, c: Column, b: Box, content:SquareNumber, isGiven:Boolean, solution:SquareNumber + associateRow(row:Unit) + associateColumn(col:Unit) + associateBox(box:Unit) + calculateCandidates(): Boolean //sets all, false if exists prior + getCandidates(): Set<ANumber> + setContent(n:ANumber) : Boolean //false if solution present and not matching n

Extends

Box

Notes:

- 1. Based on SudokuAnalyzerDomainDiagramv2
- 2. Includes only subset of classes at this time (focus on puzzle components only)
- 3. Use of <<constructor>> is alternative to CreateXyz(...) method name

Details:

- 1. ANumber 1..9 (value for square); INumber 0..8 (internal index to row, column, box arrays); only indexes are 0.., all other are 1..; opportunities for off-by-one abound?!
- 2. SquareNumber 0..9 (ANumber and zero for empty square)
- 3. Not yet sure if need Row, Column, Box or just Unit
- 4. Squares, Boxes, Rows, Columns are called and named by ANumber values (starting with 1)
- 5. Consistency is all rules met; only full Puzzle has attribute; updated after every solution square. Other components check consistency on demand (method to be specified in class diagram)
- 6. Associations between Puzzle, Unit, Square will all be implemented as final (don't changé once set up)
- 7. CandidateChange is {Square, ANumberRemoved}