Padam Chopra | December 15, 2020 FileManager Model <<enumeration>> MacroInfo Modes + searchFile: String - currentView: Integer NORMAL MODE + macrosFile: String - views [1...]: View[] INSERT MODE # controller: Controller EX MODE + getLastCharacter(): Int + readFromFile(in String): String[0...] View Controller + writeToFile(in String, in String[1...]) + createFile(in String) + isEmpty(): Boolean # addView(in View) + isValid(): Boolean + getInput(): Int[1...] + updateView(in State) + updateStatus(in String) getKevPress(): Integer + updateStatusWithError(in String) doClearStatus() - updateView(in WindowState) + getInput(): Int[1...] + update(in WindowState) + popLastInput() + updateStatus(in String) + resetInput() + updateStatusWithError(in String) Mode + clearStatus() VMModel states [1...]: WindowState[] - modes [1...]: Mode[] NormalMode - macroInfos [0...]: MacroInfo[] ExMode TerminalView Terminal parseInputSequence(in Integer[1...]) + parseInput() + start() - updateView(in WindowState) + removeActiveState() getKeyStroke(in TerminalWindow): Integer parseInputSequence(in Integer[1...]) + changeMode(in Modes) + printString(in TerminalWindow, in String) + simulateInputSequence(in Integer∏, in + clearInputSequence() parseInputSequence(in Integer[1...]) + linesOccupied(in Integer): Integer Integer) + getInput(): Integer[1...] InsertMode + startColoredOutput(in TerminalWindow) + updateStatus(in String) + stopColoredOutput(in TerminalWindow) WindowState + startRecordingSequence(in String) + startErrorOutput(in TerminalWindow) + stopRecordingSequence() + stopErrorOutput(in TerminalWindow) + playMacro(in String) - pastTimeline: History[0...] - parseInputSequence(in Integer[1...]) + clear(in TerninalWindow) + partlyEraseRecordingSequence(in Integer[]) futureTimeline: History[0...] + executeRecordedSequence(); + clear() **ExCommand** Command + addState(in String) + makeExitPreparations(); + clearSequence(); + addLine(in Integer, in String) # recordable: Boolean # commandSequence: Int[1...] + removeLine(in Integer) # countable: Boolean + saveFile(in String) # undoable: Boolean executeFor(in VMModel, in String) + memoriseState(in TerminalController) # commandSequence: Int[1...] <<enumeration>> C++ wrapper class for NCurses + isAMatch(in Int[1...]): Boolean + redoState(in Integer): String **SpecialChars** + execute(in VMModel, in Integer[1...*]) + previousState(in Integer): String - executeFor(in VMModel, in Integer) + revalidateCursorForLine() CTRL C + isAMatch(in Int[1...]): Integer ENTER + execute(in VMModel, in Integer) BACKSPACE ESC RARROW LARROW History <<CommandName>> UARROW Like MotionCommand, InsertCommand, DARROW EditingCommand, CutPasteCommand, CTRL_B MacroCommand, etc. <<CommandType>> CTRL_D CTRL G + timeDifference(): Double CTRL F Like ExLineNumber, ExW, ExQ, ExWQ, ExAuth, CTRL U ExN, ExPrev etc. CTRL R Like J, H, K, L, I, MacroQ, MacroAt, O, CapO, CapN, N, etc. <<CommandName>>

VM UML Model

TerminalController

<<enumeration>>
TerminalWindow

getKeyPress(): Integer

+ linesAsOutput(in Integer): Integer

+ positionCursorAtStatus(in Integer)

+ outputWindowHeight(): Integer

doClearStatus()

FILEWINDOW

STATUSBAR