

```

// Swift source code of StockfishEngine.swift Version 17.0 at greenchess.com //
// import needed system files //
import AppKit

// the stockfish engine class //
class StockfishEngine
{
    // the stockfish engine command //
    var StockfishEngineCommand:String = ""

    // a loop index //
    var Index:Int = 0

    // the pointer to the stockfish engine //
    var StockfishEngine:UnsafeMutableRawPointer = StockfishEngineNew()

    // initialize the stockfish engine //
    func Initialize() {StockfishInitialize(StockfishEngine)}

    // get the information string that includes the principal variation //
    func GetInfoString() -> String
    {
        // return the information string that includes the principal variation //
        return String(cString:StockfishGetInfoString(StockfishEngine))
    }

    // get the stockfish engine name //
    func GetEngineName() -> String
    {
        // return the stockfish engine name //
        return String(cString:StockfishGetEngineName(StockfishEngine))
    }

    // find the best stockfish engine move //
    func FindBestMove() -> String
    {
        // return the best stockfish engine move //
        return String(cString:StockfishFindBestMove(StockfishEngine))
    }

    // write to the stockfish engine to stop //
    func WriteStockfishToStop() {StockfishWrite(StockfishEngine, "stop\n")}

    // write to the stockfish engine to use nnue //
    func WriteStockfishUseNNUE()
    {
        // write to the stockfish engine to use nnue //
        StockfishWrite(StockfishEngine, "setoption name Use NNUE value true\n")
    }

    // write to the stockfish engine the hash value to use //
    func WriteStockfishHashSize(_ Hash:Int)
    {
        // write to the stockfish engine the hash value to use //
        StockfishWrite(StockfishEngine, "setoption name Hash value " +
            String(Hash) + "\n")
    }

    // write to the stockfish engine the skill level to use //
    func WriteStockfishSkillLevel(_ Level:Int)
    {
        // write to the stockfish engine the skill level to use //
        StockfishWrite(StockfishEngine, "setoption name Skill Level value " +
            String(Level) + "\n")
    }

    // write to the stockfish engine the search depth in plies //
    func WriteStockfishToMakeNewMove(_ Plies:Int)
    {
        // write to the stockfish engine the search depth in plies //
        StockfishWrite(StockfishEngine, "go depth " + String(Plies) + "\n")
    }

    // write to the stockfish engine to start a new game //
    func WriteStockfishNewGame()
    {
        // write to the stockfish engine to start a new game //
        StockfishWrite(StockfishEngine, "ucinewgame\n")
    }

    // write to the stockfish engine to only use one thread //
    func WriteStockfishSingleThread()
    {
        // write to the stockfish engine to only use one thread //
        StockfishWrite(StockfishEngine, "setoption name Threads value 1\n")
    }

    // write the number of threads to use for the stockfish engine //
    func WriteStockfishMaxThreads()
    {
        // set the number of threads to use for the stockfish engine //
        StockfishWrite(StockfishEngine, "setoption name Threads value " +
            String(ProcessInfo.processInfo.processorCount) + "\n")
    }

    // write the stockfish position from the current fen position string //
    func WriteStockfishPositionFromFEN(_ FENString:String, MoveList:[String])
    {
        // set to start from the current fen position string //
        StockfishEngineCommand = "position fen " + String(FENString) + " moves" +
            "\n"

        // loop over all the moves in the move list //
        for Index in 0 ..< MoveList.count
        {
            // add a move to the stockfish engine command //
            StockfishEngineCommand += " " + MoveList[Index]
        }

        // write a newline to the stockfish engine command //
        StockfishEngineCommand += "\n"

        // write the command to the stockfish engine //
        StockfishWrite(StockfishEngine, StockfishEngineCommand)
    }

    // write the stockfish position from the default fen //
    func WriteStockfishCurrentPositionFromDefaultFEN(_ MoveList:[String])
    {
        // set to start from the default fen position //
        StockfishEngineCommand = "position startpos moves"

        // loop over all the moves in the move list //
        for Index in 0 ..< MoveList.count
        {
            // add a move to the stockfish engine command //
            StockfishEngineCommand += " " + MoveList[Index]
        }

        // write a newline to the stockfish engine command //
        StockfishEngineCommand += "\n"

        // write the command to the stockfish engine //
        StockfishWrite(StockfishEngine, StockfishEngineCommand)
    }
}

```