```
2c2
< // Swift source code of main.swift Version 9.0.3 at greenchess.com //
---
> // Swift source code of main.swift Version 9.0.2 at greenchess.com //
14,26d13
<    // the list of opening books //
<    let OpeningBooks:[String] = ["eco", "CenterGame-Danish", "English1b6", "Hodgson", "OwenDefense",
<                                 "SemiBenoni"]
<
<    // the opening book dispatch queue //
<    let OpeningBookQueue = DispatchQueue(label:"OpeningBookThread", attributes:.concurrent)
<
<    // the engine dispatch queue //
<    let EngineQueue = DispatchQueue(label:"EngineThread", attributes:.concurrent)
<
<    // the sketch being drawn to //
<    var Sketch:Sketching = Sketching()
<
41a29,31
>    // the drawing diagram layer //
>    @IBOutlet var DiagramLayer:DrawingLayer!
>
153c143,146
<
---
>
>    // the diagram cursor //
>    var DiagramCursor:NSCursor = NSCursor()
>
221,223d213
<    var BookIndex:Int = 0
<
<    // an index in a list //
351a342,347
>    // the read the opening book thread //
>    var ReadOpeningBookThread:Thread = Thread()
>
>    // the engine computation thread //
>    var EngineThread:Thread = Thread()
>
720,721c716,717
<       // run the thread to read in the chess opening books //
<       OpeningBookQueue.async {self.ReadOpeningBooks()}
---
>       // run the thread to read in the chess opening book //
>       RunReadOpeningBookThread()
741,756d736
<    // undo the diagram path //
<    @IBAction func UndoPath(_:AnyObject) {GameSounds.Play("click"); Sketch.UndoPath()}
<
<    // update the diagram color //
<    @IBAction func UpdateDiagramColor(_:AnyObject)
<    {
<       // play the game sound of a click //
<       GameSounds.Play("click")
<
<       // update the sketch diagram color //
<       Sketch.UpdateColor(MainToolbar!.selectedItemIdentifier!.rawValue)
<    }
<
<    // clear the diagram the user drew on //
<    @IBAction func ClearDiagram(_:AnyObject) {GameSounds.Play("clear"); Sketch.Refresh(true)}
<
1533c1513,1516
<
---
>
>          // set the uci engine riskiness option //
>          Stockfish.WriteStockfishRiskiness(Inf.Vals("UCIEngineRiskiness").VI)
>
1589c1572
<    func ComputeEngineMove()
---
>    @objc func ComputeEngineMove()
1600a1584,1586
>          // if will be using the encyclopedia of chess openings book, wait until book read in //
>          if Inf.Vals("UseEcoBook").VB {while ReadOpeningBookThread.isExecuting {}}
>
2322a2309,2314
>       // update the uci engine riskiness slider //
>       Ctr.UCIEngineRiskinessSlider.intValue = Int32(Inf.Vals("UCIEngineRiskiness").VI)
>
>       // display the uci engine riskiness //
>       DisplayUCIEngineRiskiness()
>
2625,2626c2617,2619
<                 "MovieFPSFromSlider":[10, 0, 100], "SavedPositionSlot":[1, 1, 5], "SearchTime":
<                 [5, 5, 120], "PreferencesSelection":[20, 20, 28]])
---
>                 "UCIEngineRiskiness":[0, -100, 100], "MovieFPSFromSlider":[10, 0, 100],
>                 "SavedPositionSlot":[1, 1, 5], "SearchTime":[5, 5, 120], "PreferencesSelection":
>                 [20, 20, 28]])
2677c2670
<       AppNameView.stringValue = "GreenChess 9.0.3"
---
>       AppNameView.stringValue = "GreenChess 9.0.2"
2762,2765d2754
<
<       // initialize the sketch //
<       Sketch.Initialize(self as NSView, Width:Dims.GetBoardSize(2), Height:Dims.GetBoardSize(2),
<                         Length:Dims.GetBoardSize(2), YOffset:0, Margin:Dims.HalfBorderWidth)
2838c2827
<             Sketch.Cursor.set()
---
>             DiagramCursor.set()
2928,2932c2917,2926
<          // play the sound of a marker pen //
<          GameSounds.Play("marker")
<
<          // start to sketch a diagram //
<          Sketch.BeginPath()
---
>          // start the diagram path on the diagram layer //
>          DiagramLayer.StartPath(Event)
>
>          // perform the mouse diagram drawing //
>          window?.trackEvents(matching:[.leftMouseDragged, .leftMouseUp], timeout:NSEvent
>                              .foreverDuration, mode:RunLoop.Mode.eventTracking, handler:
>                              {(Event, Done) in guard let Event = Event else {return}
>                              if Event.type == .leftMouseDragged {UpdateCursor(Event.locationInWindow,
>                              Moving:false); DiagramLayer.DrawPath(Event)} else {DiagramLayer
>                              .EndPath(Event); Done.pointee = true}})
3082,3083c3076,3083
<       // compute the engine move //
<       EngineQueue.async {self.ComputeEngineMove()}
---
>       // create the chess engine thread //
>       EngineThread = Thread(target:self, selector:#selector(ComputeEngineMove), object:nil)
>
>       // set the chess engine thread to the highest priority //
>       EngineThread.threadPriority = 1
>
>       // start the chess engine thread //
>       EngineThread.start()
3193c3193
<    override func draw(_ theRect:CGRect)
---
>    override func draw(_ Rect:CGRect)
3195,3199c3195,3196
<       // if currently diagramming, update the sketch to the screen //
<       if Inf.Vals("Diagramming").VB {Sketch.SketchImages.UpdateScreen(); Sketch.Displaying = false}
<
<       // if currently not diagramming //
<       if !Inf.Vals("Diagramming").VB
---
>       // if diagramming //
>       if Inf.Vals("Diagramming").VB
3201c3198,3205
<          // draw the board and pieces image to the view //
---
>          // draw the diagram image into the main view //
>          NSGraphicsContext.current?.cgContext.draw(DiagramLayer.GetImage(), in:bounds, byTiling:
>                                                    false)
>       }
>       // not diagramming //
>       else
>       {
>          // draw the board and pieces image into the main board view //
3213c3217
<       GameGalleryView.Sketch.Refresh(true)
---
>       GameGalleryView.ClearDiagram()
3217,3224c3221
<    @IBAction func GalleryUndoPath(_:AnyObject)
<    {
<       // play the sound of a button click //
<       GameSounds.Play("click")
<
<       // undo a path in the gallery //
<       GameGalleryView.Sketch.UndoPath()
<    }
---
>    @IBAction func UndoGalleryPath(_:AnyObject) {GameSounds.Play("click"); GameGalleryView.Undo()}
3229,3231d3225
<       // play the game sound of a click //
<       GameSounds.Play("click")
<
3412c3406
<          Adaptor!.append(PixelBufferFromCGImage(GameGalleryView.MovieFrame(FrameIndex)),
---
>          Adaptor!.append(PixelBufferFromCGImage(GameGalleryView.MovieFrame()),
3418a3413,3415
>
>       // reset the gallery view //
>       GameGalleryView.ResetGallery()
3497,3500c3494,3497
<
<       // set the snapshot image rep //
<       SnapshotImageRep = Inf.Vals("Diagramming").VB ? NSBitmapImageRep(cgImage:Sketch.Snapshot()) :
<                          NSBitmapImageRep(cgImage:BPComp.GetCGImage(2))
---
>
>       // get the snapshot image representation from the board and pieces or the diagram image //
>       SnapshotImageRep = Inf.Vals("Diagramming").VB ? NSBitmapImageRep(cgImage:DiagramLayer
>                          .GetImage()) : NSBitmapImageRep(cgImage:BPComp.GetCGImage(2))
3615,3616c3612,3613
<    // read the opening books //
<    func ReadOpeningBooks()
---
>    // run the read the opening book thread //
>    func RunReadOpeningBookThread()
3618,3622c3615,3622
<       // initialize the chess opening book //
<       OpeningBookInitializeAttributes(OpeningBook)
<
<       // loop over the list of opening books //
<       for BookIndex in 0 ..< OpeningBooks.count {ReadOpeningBook(OpeningBooks[BookIndex])}
---
>       // create the thread to read in the opening book //
>       ReadOpeningBookThread = Thread(target:self, selector:#selector(ReadOpeningBook), object:nil)
>
>       // set the read opening book thread priority to the maximum value //
>       ReadOpeningBookThread.threadPriority = 1
>
>       // start the read opening book thread //
>       ReadOpeningBookThread.start()
3626c3626
<    func ReadOpeningBook(_ OpeningBookName:String)
---
>    @objc func ReadOpeningBook()
3627a3628,3633
>       // clear the opening book move list //
>       OpeningBookMoveList.removeAll()
>
>       // initialize the chess opening book //
>       OpeningBookInitializeAttributes(OpeningBook)
>
3629,3630c3635,3636
<       DatabaseReadGameCollection(OpeningBookGameList, Bundle.main.path(forResource:OpeningBookName,
<                                  ofType:"pgn")!)
---
>       DatabaseReadGameCollection(OpeningBookGameList, Bundle.main.path(forResource:"eco", ofType:
>                                  "pgn")!)
3643,3644c3649,3650
<          // loop over the move list for this game in the opening book up to the first 10 moves //
<          for MoveListCountIndex in 0 ..< min(10, OpeningBookMoveList.count)
---
>          // loop over the move list for this game in the opening book //
>          for MoveListCountIndex in 0 ..< OpeningBookMoveList.count
4071,4075c4077,4078
<       // set the main toolbar selected item to 309 for yellow color //
<       MainToolbar.selectedItemIdentifier = NSToolbarItem.Identifier(rawValue:"309")
<
<       // set the diagram cursor to yellow highlighter //
<       Sketch.UpdateColor("309")
---
>       // set the main toolbar selected item to yellow color //
>       DiagramLayer.SetMarkerToYellow()
4082,4084d4084
<
<       // refresh the sketch //
<       Sketch.Refresh(true)
4290,4291c4290,4294
<       // update the current position in the gallery //
<       GameGalleryView.UpdateCurrentPosition(Int(GameGallerySlider.intValue))
---
>       // set the current position to the selected one from the game gallery slider //
>       GameGalleryView.CurrentPosition = Int(GameGallerySlider.intValue)
>
>       // update the game gallery view //
>       GameGalleryView.UpdateGallery()
4336a4340,4342
>
>       // update the gallery //
>       GameGalleryView.UpdateGallery()
4338,4341c4344
<       // refresh the sketch to the screen //
<       GameGalleryView.Sketch.Refresh(true)
<
<       // open the game gallery window //
---
>       // begin the game gallery window //
5081,5085c5084,5104
<       // if currently analyzing the game, reset the game data //
<       if Inf.Vals("AnalyzingGame").VB {Inf.TurnOffVB("VB", Key:"AnalyzingGame"); ResetGameData()}
<
<       // if currently diagramming, reset the game data //
<       if Inf.Vals("Diagramming").VB {Inf.TurnOffVB("VB", Key:"Diagramming"); ResetGameData()}
---
>       // if already currently analyzing the game //
>       if Inf.Vals("AnalyzingGame").VB
>       {
>          // set analyzing game to be false //
>          Inf.TurnOffVB("VB", Key:"AnalyzingGame")
>
>          // reset the game data //
>          ResetGameData()
>       }
>       // if currently diagramming //
>       else if Inf.Vals("Diagramming").VB
>       {
>          // set that no longer in diagram mode //
>          Inf.TurnOffVB("VB", Key:"Diagramming")
>
>          // reset the diagram later //
>          DiagramLayer.ResetDiagram(false)
>
>          // reset the game data //
>          ResetGameData()
>       }
5408a5428,5438
>
>    // update the uci engine riskiness //
>    @IBAction func UpdateUCIEngineRiskiness(_:AnyObject)
>    {
>       // get the current uci engine riskiness from the uci engine riskiness slider //
>       Inf.Update("VIPref", VIItems:["UCIEngineRiskiness":Int(Ctr.UCIEngineRiskinessSlider
>                 .intValue)])
>
>       // display the uci engine riskiness //
>       DisplayUCIEngineRiskiness()
>    }
5503a5534,5540
>    // display the uci engine riskiness //
>    func DisplayUCIEngineRiskiness()
>    {
>       // display the uci engine riskiness //
>       Ctr.UCIEngineRiskinessView.stringValue = "Riskiness is " + Inf.IntStr("UCIEngineRiskiness")
>    }
>
6044a6082,6084
>       // invalidate the gallery diagram path //
>       GameGalleryView.Inf.TurnOffVB("VB", Key:"ValidPath")
>
6072c6112,6115
<
---
>
>       // update the game gallery view cursor //
>       GameGalleryView.UpdateCursor()
>
6548,6552c6591,6592
<       // if diagramming, update the sketch //
<       if Inf.Vals("Diagramming").VB
<       {
<          // set the sketch background //
<          Sketch.SetBackground(BPComp.GetCGImage(2))
---
>       // if diagramming, update the diagram context //
>       if Inf.Vals("Diagramming").VB {DiagramLayer.Refresh(!Inf.Vals("EnteringPreferences").VB)}
6554,6556d6593
<          // refresh the sketch //
<          Sketch.Refresh(false)
<       }
6558,6562c6595
<       else if Display
<       {
<          // update the display //
<          display(bounds)
<       }
---
>       if Display {setNeedsDisplay(bounds)}
```