```
2c2
< // Swift source code of main.swift Version 9.0.8 at greenchess.com //
---
> // Swift source code of main.swift Version 9.0.7 at greenchess.com //
10,12d9
< // the public user interface //
< public var GCUI:GCUserInterface = GCUserInterface()
<
18,20d14
<    // the grid rows list //
<    var GridRows:[String] = ["Wraparound 2 rows of boards in Grid View",
<                            "Wraparound 3 rows of boards in Grid View"]
23a18,20
>    // the user interface //
>    var UI:UserInterface = UserInterface()
>
25,28c22,28
<    var PrefsSubWindows:[Int:String] = [1:"PrefsBackground", 2:"PrefsBoard", 3:"PrefsClock",
<                                       4:"PrefsDiagram", 5:"PrefsFont", 6:"PrefsGallery",
<                                       7:"PrefsMovie", 8:"PrefsSpeed", 9:"PrefsStockfish"]
<
---
>    var PrefsSubWindows:[Int:String] = [20:"PrefsBackground", 21:"PrefsBoard", 22:"PrefsFont",
>                                       23:"PrefsStockfish", 24:"PrefsSpeed", 25:"PrefsClock",
>                                       26:"PrefsMovie", 27:"PrefsGallery", 28:"PrefsDiagram"]
>
>    // the frames movie creator //
>    @IBOutlet var FramesMovie:FramesMovieCreator!
>
65a66,68
>    // the preference controller //
>    @IBOutlet var Ctr:PreferenceController!
>
68c71,74
<
---
>
>    // the game attributes //
>    @IBOutlet var Attributes:GameAttributes!
>
80c86,89
<
---
>
>    // the board and piece compositor //
>    @IBOutlet var BPComp:BoardPieceCompositor!
>
326c335,338
<
---
>
>    // the game sounds //
>    @IBOutlet var GameSounds:GameSoundEffects!
>
350c362,365
<
---
>
>    // the pgn moves list data //
>    @IBOutlet var PGNMovesData:PGNMovesDataSource!
>
362a378,380
>    // the preferences toolbar //
>    @IBOutlet var PreferencesToolbar:NSToolbar!
>
478c496
<    var PreferencesWindow:AWindow = AWindow()
---
>    @IBOutlet var PreferencesWindow:AWindow!
618c636,639
<
---
>
>    // the previous preferences selection //
>    var PreviousPreferencesSelection:Int = 0
>
665c686
<    @IBAction func UndoPath(_:AnyObject) {GCUI.GameSounds.Play("click"); Sketch.UndoPath()}
---
>    @IBAction func UndoPath(_:AnyObject) {GameSounds.Play("click"); Sketch.UndoPath()}
671c692
<       GCUI.GameSounds.Play("click")
---
>       GameSounds.Play("click")
678c699
<    @IBAction func ClearDiagram(_:AnyObject) {GCUI.GameSounds.Play("clear"); Sketch.Refresh(true)}
---
>    @IBAction func ClearDiagram(_:AnyObject) {GameSounds.Play("clear"); Sketch.Refresh(true)}
690c711
<       Inf.Update("VIPref", VIItems:["WindowBackgroundHue":GCUI.GetColorView("WindowColorView")
---
>       Inf.Update("VIPref", VIItems:["WindowBackgroundHue":Ctr.WindowBackgroundColorView
694c715
<       MainWindow.backgroundColor = WindowTexture.GetTexture(GCUI.GetColorView("WindowColorView")
---
>       MainWindow.backgroundColor = WindowTexture.GetTexture(Ctr.WindowBackgroundColorView
701c722
<       WhiteTimeClock.UpdateClockWindowColor(GCUI.GetColorView("WindowColorView").GetCurrentHue())
---
>       WhiteTimeClock.UpdateClockWindowColor(Ctr.WindowBackgroundColorView.GetCurrentHue())
704c725
<       BlackTimeClock.UpdateClockWindowColor(GCUI.GetColorView("WindowColorView").GetCurrentHue())
---
>       BlackTimeClock.UpdateClockWindowColor(Ctr.WindowBackgroundColorView.GetCurrentHue())
707,708c728
<       GCUI.GetClockView("ClockView").UpdateClockWindowColor(GCUI.GetColorView("WindowColorView")
<                                                             .GetCurrentHue())
---
>       Ctr.TimeClockPreview.UpdateClockWindowColor(Ctr.WindowBackgroundColorView.GetCurrentHue())
806c826
<             .Vals("IllustrationSize").VI), UpdateBoard:false, Table:true), GCUI.PGNMovesData
---
>             .Vals("IllustrationSize").VI), UpdateBoard:false, Table:true), PGNMovesData
834c854
<                                                      .SelectedRow(Row, GameListFont:GameListFont) :
---
>                                                      .SelectedRow(Row,GameListFont:GameListFont) :
849,851c869,871
<       return (TableView == PGNMovesTableView) ? min(GCUI.PGNMovesData.RowHeight(Row), 2000) :
<              ((TableView == GameInfoTableView) ? GameInfoData.RowHeight(Row) : ((TableView ==
<              GameListTableView) ? GameListData.RowHeight(Row) : 30))
---
>       return (TableView == PGNMovesTableView) ? min(PGNMovesData.RowHeight(Row), 2000) : ((TableView
>              == GameInfoTableView) ? GameInfoData.RowHeight(Row) : ((TableView == GameListTableView)
>              ? GameListData.RowHeight(Row) : 30))
893,895d912
<       // make the app active //
<       NSApp.activate(ignoringOtherApps:true)
<
1002c1019
<          GCUI.Attributes.Inf.SaveNotation("Moves", VNS:String(validatingUTF8:BoardPGNLabel(Board))!)
```

```
---
>          Attributes.Inf.SaveNotation("Moves", VNS:String(validatingUTF8:BoardPGNLabel(Board))!)
1031c1048
<       GCUI.GameSounds.Play("thump")
---
>       GameSounds.Play("thump")
1037c1054
<       UpdateIllustrationData(GCUI.Attributes.Inf.GetCount())
---
>       UpdateIllustrationData(Attributes.Inf.GetCount())
1074,1075c1091,1092
<    // update the display game clock toggle button //
<    func UpdateDisplayGameClock(_ Key:String)
---
>    // display the game clock in panel action //
>    @IBAction func DisplayGameClockInPanelButtonAction(_:AnyObject)
1077,1083c1094,1099
<       // toggle the hide game clock in panel button //
<       Inf.ToggleVB("VBPref", Key:Key)
<
<       // update the time clock hidden status //
<       (WhiteTimeClock.isHidden, BlackTimeClock.isHidden) = (Inf.Vals("HideGameClockInPanel").VB,
<       Inf.Vals("HideGameClockInPanel").VB)
<
---
>       // toggle the hide game clock in panel //
>       TogglePref("HideGameClockInPanel")
>
>       // hide or unhide the game clock in the panel //
>       HideGameClockInPanelAction()
>
1087a1104,1111
>    // hide the game clock in panel action //
>    func HideGameClockInPanelAction()
>    {
>       // update the time clock hidden states //
>       (WhiteTimeClock.isHidden, BlackTimeClock.isHidden) = (Inf.Vals("HideGameClockInPanel").VB,
>       Inf.Vals("HideGameClockInPanel").VB)
>    }
>
1131,1132c1155,1156
<       InFullScreen, frame) = (UpdatedState(Inf.Vals("HideInfoPanel").VB), UpdatedState(Inf
<       .Vals("LargeInfoPanel").VB), UpdatedState(Inf.Vals("SmallInfoPanel").VB), BoolTo().CG(Inf
---
>       InFullScreen, frame) = (Ctr.UpdatedState(Inf.Vals("HideInfoPanel").VB), Ctr.UpdatedState(Inf
>       .Vals("LargeInfoPanel").VB), Ctr.UpdatedState(Inf.Vals("SmallInfoPanel").VB), BoolTo().CG(Inf
1162c1186
<       PGNMovesTableView.scrollRowToVisible(GCUI.PGNMovesData.LastRow())
---
>       PGNMovesTableView.scrollRowToVisible(PGNMovesData.LastRow())
1263,1274c1287,1288
<       // if moving by selecting squares and have selected square //
<       if GCUI.BPComp.SelectedSquare
<       {
<          // reset the moving piece //
<          MovingPiece.Reset()
<
<          // initialize the selected square //
<          GCUI.BPComp.InitSelectedSquare()
<
<          // refresh all //
<          Refresh("All")
<       }
---
>       // if moving by selecting squares and have selected squares //
>       if BPComp.SelectedSquare {MovingPiece.Reset(); BPComp.InitSelectedSquare(); Refresh("All")}
1348,1350c1362,1364
<       Stockfish.WriteStockfishPositionFromFEN(GCUI.Attributes.FENString(), MoveList:GCUI.Attributes
<                                               .Inf.GetVSList("Moves"))
<
---
>       Stockfish.WriteStockfishPositionFromFEN(Attributes.FENString(), MoveList:Attributes.Inf
>                                               .GetVSList("Moves"))
>
1389,1390c1403
<       if GCUI.Attributes.NoMoves() && GCUI.Attributes.IsDefaultFENAndWhite() && !Inf
<          .Vals("UseEcoBook").VB
---
>       if Attributes.NoMoves() && Attributes.IsDefaultFENAndWhite() && !Inf.Vals("UseEcoBook").VB
1420c1433
<       GCUI.Attributes.Inf.SaveNextVS("Moves", VS:MovingPiece.UCIMove)
---
>       Attributes.Inf.SaveNextVS("Moves", VS:MovingPiece.UCIMove)
1446c1459
<       if GCUI.Attributes.Inf.RedoToNextVS() {Refresh("All"); StoreAutosaveData("")}
---
>       if Attributes.Inf.RedoToNextVS() {Refresh("All"); StoreAutosaveData("")}
1489c1502
<       GCUI.GameSounds.Play(Sound)
---
>       GameSounds.Play(Sound)
1514c1527
<       FrameNSImage(GCUI.Attributes.Inf.GetCount(), Size:100, UpdateBoard:false, Table:false) : (Inf
---
>       FrameNSImage(Attributes.Inf.GetCount(), Size:100, UpdateBoard:false, Table:false) : (Inf
1537,1541c1550
<       for Clock in [WhiteTimeClock, BlackTimeClock, GCUI.GetClockView("ClockView")]
<       {
<          // set that need to refresh the entire clock //
<          Clock!.RefreshAll = true
<       }
---
>       for Clock in [WhiteTimeClock, BlackTimeClock, Ctr.TimeClockPreview] {Clock!.RefreshAll = true}
1551c1560
<       if GCUI.GetClockView("ClockView").RefreshAll {GCUI.GetClockView("ClockView").UpdateClock(0)}
---
>       if Ctr.TimeClockPreview.RefreshAll {Ctr.TimeClockPreview.UpdateClock(0)}
1615c1624
<             if (GameState == 5) || (GameState == 6) {GCUI.GameSounds.Play("warning")}
---
>             if (GameState == 5) || (GameState == 6) {GameSounds.Play("warning")}
1662c1671
<       GCUI.BPComp.InitSelectedSquare()
---
>       BPComp.InitSelectedSquare()
1678c1687
<       GCUI.Attributes.Initialize()
---
>       Attributes.Initialize()
1689c1698
<       // refresh all the views //
---
>       // refresh all //
1719c1728
<       if GCUI.Attributes.Inf.Vals("FENEnpassantSquare").VS.count == 2
---
>       if Attributes.Inf.Vals("FENEnpassantSquare").VS.count == 2
1722,1723c1731,1732
<          (File, Rank) = (SquareLabel.FileIndex(GCUI.Attributes.Inf.Vals("FENEnpassantSquare").VS
<          .first!), SquareLabel.RankIndex(GCUI.Attributes.Inf.Vals("FENEnpassantSquare").VS.last!))
```

```
---
>           (File, Rank) = (SquareLabel.FileIndex(Attributes.Inf.Vals("FENEnpassantSquare").VS.first!),
>           SquareLabel.RankIndex(Attributes.Inf.Vals("FENEnpassantSquare").VS.last!))
1730,1732c1739,1740
<                 1, 0) && GCUI.Attributes.BlackStarts()) || ((Rank == 5) && BPData.RFE(File, 4, -1)
<                 && BPData.RFE(File, 5, 0) && BPData.RFE(File, 6, 0) && GCUI.Attributes
<                 .WhiteStarts())
---
>                 1, 0) && (Attributes.BlackStarts())) || ((Rank == 5) && BPData.RFE(File, 4, -1) &&
>                 BPData.RFE(File, 5, 0) && BPData.RFE(File, 6, 0) && (Attributes.WhiteStarts()))
1738,1739c1746,1747
<                 EnpassantSquare.Set(File, Y:Rank, Dims:Dims, Rotated:GCUI.BPComp.Inf
<                                     .Vals("RotateBoard").VB)
---
>                 EnpassantSquare.Set(File, Y:Rank, Dims:Dims, Rotated:BPComp.Inf.Vals("RotateBoard")
>                                     .VB)
1748c1756
<           GCUI.Attributes.Inf.Update("FENPref", VSItems:["FENEnpassantSquare":"-"])
---
>           Attributes.Inf.Update("FENPref", VSItems:["FENEnpassantSquare":"-"])
1818,1821c1826,1828
<        BoardInitializeAttributes(InitializingBoard, GCUI.Attributes.Inf.Vals("FENLayout").VS, GCUI
<                                  .Attributes.Inf.Vals("FENCastlingRights").VS, GCUI.Attributes.Inf
<                                  .Vals("FENColor").VS, GCUI.Attributes.Inf.Vals("FENEnpassantSquare")
<                                  .VS)
---
>        BoardInitializeAttributes(InitializingBoard, Attributes.Inf.Vals("FENLayout").VS, Attributes
>                                  .Inf.Vals("FENCastlingRights").VS, Attributes.Inf.Vals("FENColor")
>                                  .VS, Attributes.Inf.Vals("FENEnpassantSquare").VS)
1825c1832
<    func NumPiecesOnBoard() -> Int
---
>    func TotalPiecesOnBoard() -> Int
1841c1848
<        if NumPiecesOnBoard() == 0 {ExitPositionEditor(); GCUI.GameSounds.Play("buttonclick"); return}
---
>        if TotalPiecesOnBoard() == 0 {ExitPositionEditor(); GameSounds.Play("buttonclick"); return}
1864c1871
<        GCUI.Attributes.Inf.ClearVS("FENPref", Key:"FENLayout")
---
>        Attributes.Inf.ClearVS("FENPref", Key:"FENLayout")
1885c1892
<                 GCUI.Attributes.Inf.AddTo("FENPref", Key:"FENLayout", AddOn:"\(Count)")
---
>                 Attributes.Inf.AddTo("FENPref", Key:"FENLayout", AddOn:"\(Count)")
1895,1896c1902
<                 GCUI.Attributes.Inf.AddTo("FENPref", Key:"FENLayout", AddOn:FENCharacters[PieceType
<                                         + 6])
---
>                 Attributes.Inf.AddTo("FENPref", Key:"FENLayout", AddOn:FENCharacters[PieceType + 6])
1903c1909
<                 GCUI.Attributes.Inf.AddTo("FENPref", Key:"FENLayout", AddOn:"/")
---
>                 Attributes.Inf.AddTo("FENPref", Key:"FENLayout", AddOn:"/")
1915,1918c1921,1930
<         // initialize the control buttons and time clock hidden status //
<         (WhiteTimeClock.isHidden, BlackTimeClock.isHidden, GCUI.GameSounds.SoundOff) = (Inf
<         .Vals("HideGameClockInPanel").VB, Inf.Vals("HideGameClockInPanel").VB, Inf.Vals("SoundOff")
<         .VB)
---
>         // initilize the slider and buttons //
>         (Ctr.DisplayIllustrationsButton.state, GameSounds.SoundOff, Ctr.GameSoundsButton.state, Ctr
>         .ShowStockfishAnalysisButton.state, Ctr.DisplayClockButton.state, Ctr.EcoBookButton.state,
>         Ctr.AutosaveGameButton.state, Ctr.MultithreadUCIEngineButton.state, PreferencesToolbar
>         .selectedItemIdentifier) = (Ctr.UpdatedState(!Inf.Vals("HideIllustrations").VB), Inf
>         .Vals("SoundOff").VB, Ctr.UpdatedState(!Inf.Vals("SoundOff").VB), Ctr.UpdatedState(Inf
>         .Vals("ShowStockfishAnalysis").VB), Ctr.UpdatedState(!Inf.Vals("HideGameClockInPanel")
>         .VB), Ctr.UpdatedState(Inf.Vals("UseEcoBook").VB), Ctr.UpdatedState(Inf.Vals("AutosaveGame")
>         .VB), Ctr.UpdatedState(Inf.Vals("MultithreadUCIEngine").VB), NSToolbarItem
>         .Identifier(rawValue:Inf.IntStr("PreferencesSelection")))
1921,1922c1933,1936
<         MoveListFont.Read(27, TheScale:1, VertOffset:Inf.Vals("IllustrationSize").VI, UseVertOffset:
<                          !Inf.Vals("HideIllustrations").VB, Tag:"MoveList")
---
>         MoveListFont.Read(Ctr.MoveListColorView, DefaultHue:27, TheFont:Ctr.MoveListFontButton, Sizes:
>                     Ctr.MoveListSizesButton, Bold:Ctr.MoveListBoldButton, TheScale:1, VertOffset:Inf
>                     .Vals("IllustrationSize").VI, UseVertOffset:!Inf.Vals("HideIllustrations").VB,
>                     Tag:"MoveList")
1925c1939,1941
<         GameInfoFont.Read(27, TheScale:1, VertOffset:0, UseVertOffset:false, Tag:"GameInfo")
---
>         GameInfoFont.Read(Ctr.GameInfoColorView, DefaultHue:27, TheFont:Ctr.GameInfoFontButton,
>                     Sizes:Ctr.GameInfoSizesButton, Bold:Ctr.GameInfoBoldButton, TheScale:1,
>                     VertOffset:0, UseVertOffset:false, Tag:"GameInfo")
1928c1944,1946
<         GameListFont.Read(27, TheScale:1, VertOffset:0, UseVertOffset:false, Tag:"GameList")
---
>         GameListFont.Read(Ctr.GameListColorView, DefaultHue:27, TheFont:Ctr.GameListFontButton,
>                     Sizes:Ctr.GameListSizesButton, Bold:Ctr.GameListBoldButton, TheScale:1,
>                     VertOffset:0, UseVertOffset:false, Tag:"GameList")
1931c1949
<         GCUI.GetColorView("WindowColorView").Initialize(Inf.Vals("WindowBackgroundHue").VI)
---
>         Ctr.WindowBackgroundColorView.Initialize(Inf.Vals("WindowBackgroundHue").VI)
1937c1955
<         GCUI.BPComp.Read(WindowTexture)
---
>         BPComp.Read(WindowTexture)
1940c1958
<         GCUI.FramesMovie.ReadPrefs()
---
>         FramesMovie.ReadPrefs()
1943c1961
<         GCUI.GetColorView("ClockColorView").Initialize(Inf.Vals("GameClockHue").VI)
---
>         Ctr.ClockColorView.Initialize(Inf.Vals("GameClockHue").VI)
1949c1967
<         GCUI.GetColorView("ClockSurfaceColorView").Initialize(Inf.Vals("GameClockBackgroundHue").VI)
---
>         Ctr.ClockBackgroundColorView.Initialize(Inf.Vals("GameClockBackgroundHue").VI)
1956a1975,1980
>         // set the current number of grid rows in the grid rows button //
>         Ctr.GridRowsButton.selectItem(at:Inf.Vals("NumberOfGridRows").VI - 2)
>
>         // update the game clock in panel hidden state //
>         HideGameClockInPanelAction()
>
1958c1982
<         ProgressCircle.UpdateCircleColor(GCUI.GetColorView("WindowColorView").GetCurrentHue())
---
>         ProgressCircle.UpdateCircleColor(Ctr.WindowBackgroundColorView.GetCurrentHue())
1964c1988
<         for PositionListIndex in 1...25 {if !GCUI.Attributes.Initialize("\(PositionListIndex)")
---
>         for PositionListIndex in 1...25 {if !Attributes.Initialize("\(PositionListIndex)")
1977c2001
<         _ = GCUI.Attributes.Initialize("")
---
>         _ = Attributes.Initialize("")
1987c2011
<         GCUI.Attributes.SaveSubset(GCUI.Attributes.Inf.GetCount())
---
```

```
>       Attributes.SaveSubset(Attributes.Inf.GetCount())
1999,2000c2023,2024
<          PositionImages.append(FrameCGImage(GCUI.Attributes.Inf.GetCount(), Size:900, UpdateBoard:
<                                 false, Table:false))
---
>          PositionImages.append(FrameCGImage(Attributes.Inf.GetCount(), Size:900, UpdateBoard:false,
>                                 Table:false))
2019c2043
<       GCUI.Attributes.StoreAutosaveData(Suffix)
---
>       Attributes.StoreAutosaveData(Suffix)
2059c2083
<       if !Inf.Vals("ValidFENString").VB {GCUI.Attributes.SetFENDefaults(); ComputeLayout()}
---
>       if !Inf.Vals("ValidFENString").VB {Attributes.SetFENDefaults(); ComputeLayout()}
2062c2086
<       if !FENLayoutIsLegal() {GCUI.Attributes.SetFENDefaults(); ComputeLayout()}
---
>       if !FENLayoutIsLegal() {Attributes.SetFENDefaults(); ComputeLayout()}
2081c2105
<       for Index in 0..<GCUI.Attributes.Inf.Vals("FENLayout").VS.count
---
>       for Index in 0..<Attributes.Inf.Vals("FENLayout").VS.count
2094c2118
<          FENPiece = GCUI.Attributes.Inf.GetC("FENLayout", Index:Index)
---
>          FENPiece = Attributes.Inf.GetC("FENLayout", Index:Index)
2134c2158
<       GCUI.Attributes.CheckFENStartMoveNumber()
---
>       Attributes.CheckFENStartMoveNumber()
2140c2164
<       GCUI.Attributes.CheckColorToMove()
---
>       Attributes.CheckColorToMove()
2151c2175
<       for Index in 0..<GCUI.Attributes.Inf.Vals("FENCastlingRights").VS.count
---
>       for Index in 0..<Attributes.Inf.Vals("FENCastlingRights").VS.count
2154c2178
<          switch GCUI.Attributes.Inf.GetC("FENCastlingRights", Index:Index)
---
>          switch Attributes.Inf.GetC("FENCastlingRights", Index:Index)
2177,2182d2200
<       // initialize the preferences window //
<       PreferencesWindow.Update(660, Height:685)
<
<       // add the colors and clocks views to the screen //
<       GCUI.AddNSViews(self, PGNTags:PGNTags)
<
2210,2218c2228,2235
<       Inf.Update("VIPref", VDefMinMaxIntItems:["WindowBackgroundHue":[64, 0, GCUI
<           .GetColorView("WindowColorView").MaxHue()], "WindowTextureOpacity":[0, 0, 100],
<           "IllustrationSize":[200, 150, 250], "GameClockHue":[27, 0, GCUI
<           .GetColorView("ClockColorView").MaxHue()], "GameClockBackgroundHue":[82, 0, GCUI
<           .GetColorView("ClockSurfaceColorView").MaxHue()], "ClockBackgroundVisibility":[30, 0,
<           100], "NumberOfGridRows":[3, 2, 3], "AnimationRate":[75, 1, 100], "GalleryAnimationRate":
<           [75, 1, 100], "UCIEngineDifficulty":[20, 0, 20], "UCIEngineHash":[16, 16, 512],
<           "SavedPositionSlot":[1, 1, 5], "SearchTime":[5, 5, 120], "PreferencesSelection":[1, 1,
<           9]])
---
>       Inf.Update("VIPref", VDefMinMaxIntItems:["WindowBackgroundHue":[64, 0, Ctr
>           .WindowBackgroundColorView.MaxHue()], "WindowTextureOpacity":[0, 0, 100],
>           "IllustrationSize":[200, 150, 250], "GameClockHue":[27, 0, Ctr.ClockColorView.MaxHue()],
>           "GameClockBackgroundHue":[82, 0, Ctr.ClockBackgroundColorView.MaxHue()],
>           "ClockBackgroundVisibility":[30, 0, 100], "NumberOfGridRows":[3, 2, 3], "AnimationRate":
>           [75, 1, 100], "GalleryAnimationRate":[75, 1, 100], "UCIEngineDifficulty":[20, 0, 20],
>           "UCIEngineHash":[16, 16, 512], "SavedPositionSlot":[1, 1, 5], "SearchTime":[5, 5, 120],
>           "PreferencesSelection":[20, 20, 28]])
2220,2222c2237,2242
<       // set windows to allow concurrent view drawing //
<       (MainWindow.allowsConcurrentViewDrawing, GameGalleryWindow.allowsConcurrentViewDrawing) =
<       (true, true)
---
>       // loop over all the windows //
>       for TheWindow in [MainWindow, GameGalleryWindow, PreferencesWindow]
>       {
>          // set that can draw concurrently //
>          TheWindow!.allowsConcurrentViewDrawing = true
>       }
2223a2244,2246
>       // initialize the preference controls //
>       Ctr.Initialize()
>
2227c2250
<
---
>
2234c2257
<       .stringValue) = (GCUI.PGNMovesData, GameListData, GameInfoData, NSColor.white
---
>       .stringValue) = (PGNMovesData, GameListData, GameInfoData, NSColor.white
2236,2238c2259,2261
<       "Hide", "Timed Game", GCUI.BPComp.Highlights.EnpassantSquareNSImage, CGPoint(x:Dims.Square
<       .Width * 0.5, y:Dims.Square.Width * 0.5), #selector(LoadGameFromListButtonAction(_:)), NSFont
<       .systemFont(ofSize:20), .justified, "GreenChess 9.0.8")
---
>       "Hide", "Timed Game", BPComp.Highlights.EnpassantSquareNSImage, CGPoint(x:Dims.Square.Width *
>       0.5, y:Dims.Square.Width * 0.5), #selector(LoadGameFromListButtonAction(_:)), NSFont
>       .systemFont(ofSize:20), .justified, "GreenChess 9.0.7")
2242a2266,2268
>       // remove all the number of grid rows items from the grid rows button //
>       Ctr.GridRowsButton.removeAllItems()
>
2251a2278,2284
>
>       // loop over the number of grid row sizes //
>       for Index in 2...GridSizes.count
>       {
>          // add the title to the grid rows button //
>          Ctr.GridRowsButton.addItem(withTitle:"Wraparound \(Index) rows of boards in Grid View")
>       }
2269c2302
<       GCUI.GetClockView("ClockView").Initialize()
---
>       Ctr.TimeClockPreview.Initialize()
2272,2273c2305,2306
<       MovingPiece.Initialize()
<
---
>       MovingPiece.Initialize(self, BPComp:BPComp, Delegate:self)
>
2301a2335,2337
>
>       // initialize the user interface ojbects //
>       UI.Initialize(self)
2305,2307d2340
<
<       // add the swiftui views to the screen //
<       GCUI.AddSwiftUIViews()
2353,2354c2386,2387
<          GCUI.Attributes.Inf.Update("FENPref", VSItems:["FENEnpassantSquare":MovingPiece
<                                     .GetSquareLabel()])
---
```

```
>          Attributes.Inf.Update("FENPref", VSItems:["FENEnpassantSquare":MovingPiece
>                               .GetSquareLabel()])
2366c2399
<            if GCUI.Attributes.Inf.VSIsEqual("FENEnpassantSquare", VS:MovingPiece.GetSquareLabel())
---
>            if Attributes.Inf.VSIsEqual("FENEnpassantSquare", VS:MovingPiece.GetSquareLabel())
2369c2402
<               GCUI.Attributes.Inf.Update("FENPref", VSItems:["FENEnpassantSquare":"-"])
---
>               Attributes.Inf.Update("FENPref", VSItems:["FENEnpassantSquare":"-"])
2373c2406
<            GCUI.GameSounds.Play("pop")
---
>            GameSounds.Play("pop")
2379c2412
<            GCUI.GameSounds.Play("thump")
---
>            GameSounds.Play("thump")
2420c2453
<         GCUI.GameSounds.Play("marker")
---
>         GameSounds.Play("marker")
2436c2469
<       if !GCUI.BPComp.SelectedSquare
---
>       if !BPComp.SelectedSquare
2448c2481
<         GCUI.GameSounds.Play("click")
---
>         GameSounds.Play("click")
2455c2488
<       if GCUI.BPComp.Inf.Vals("MoveBySelectingSquares").VB
---
>       if BPComp.Inf.Vals("MoveBySelectingSquares").VB
2458c2491
<         if !GCUI.BPComp.SelectedSquare
---
>         if !BPComp.SelectedSquare
2461c2494
<            GCUI.BPComp.SetSelectedSquare()
---
>            BPComp.SetSelectedSquare()
2473c2506
<            GCUI.BPComp.InitSelectedSquare()
---
>            BPComp.InitSelectedSquare()
2541c2574
<       BPData.UpdateStartAndEndSquareForMove(MovingPiece.UCIMove, Dims:Dims, Rotated:GCUI.BPComp.Inf
---
>       BPData.UpdateStartAndEndSquareForMove(MovingPiece.UCIMove, Dims:Dims, Rotated:BPComp.Inf
2549c2582
<         GCUI.GameSounds.Play("wrongsquare")
---
>         GameSounds.Play("wrongsquare")
2588c2621
<       GCUI.Attributes.Inf.SaveNextVS("Moves", VS:MovingPiece.UCIMove)
---
>       Attributes.Inf.SaveNextVS("Moves", VS:MovingPiece.UCIMove)
2597c2630
<       GCUI.Attributes.Inf.SaveNotation("Moves", VNS:String(validatingUTF8:BoardPGNLabel(Board))!)
---
>       Attributes.Inf.SaveNotation("Moves", VNS:String(validatingUTF8:BoardPGNLabel(Board))!)
2603c2636
<       UpdateIllustrationData(GCUI.Attributes.Inf.GetCount())
---
>       UpdateIllustrationData(Attributes.Inf.GetCount())
2609c2642
<       if !Inf.Vals("QueryPawnPromotion").VB {GCUI.GameSounds.Play("thump")}
---
>       if !Inf.Vals("QueryPawnPromotion").VB {GameSounds.Play("thump")}
2618,2619c2651,2652
<    // update the stockfish analysis toggle button //
<    func UpdateStockfishAnalysis(_ Key:String) {Inf.ToggleVB("VBPref", Key:Key); Refresh("All")}
---
>    // the show the stockfish analysis button action //
>    @IBAction func StockfishAnal(_:AnyObject) {TogglePref("ShowStockfishAnalysis"); Refresh("All")}
2621,2622c2654,2655
<    // update if will have game sounds off //
<    func UpdateGameSoundsOff(_ Key:String)
---
>    // the game sounds button action //
>    @IBAction func GameSoundsButtonAction(_:AnyObject)
2624,2625c2657,2658
<       // update the games sounds off option //
<       Inf.ToggleVB("VBPref", Key:Key)
---
>       // switch the sound mode //
>       TogglePref("SoundOff")
2628c2661
<       GCUI.GameSounds.SoundOff = Inf.Vals("SoundOff").VB
---
>       GameSounds.SoundOff = Inf.Vals("SoundOff").VB
2658c2691
<       GCUI.GameSounds.Play("clear")
---
>       GameSounds.Play("clear")
2668c2701
<       GCUI.GameSounds.Play("click")
---
>       GameSounds.Play("click")
2678c2711
<       GCUI.GameSounds.Play("click")
---
>       GameSounds.Play("click")
2688c2721
<       GCUI.GameSounds.Play("camera")
---
>       GameSounds.Play("camera")
2709c2742
<       SavePanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in GCUI.GameSounds
---
>       SavePanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in self.GameSounds
2719c2752
<       GCUI.GameSounds.Play("movie")
---
>       GameSounds.Play("movie")
2740c2773
<       SavePanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in GCUI.GameSounds
---
>       SavePanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in self.GameSounds
2753c2786
<       SavePanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in GCUI.GameSounds
---
>       SavePanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in self.GameSounds
2755c2788
<                 .orderOut(nil); GCUI.FramesMovie.WriteMovie(self.Graphics.CreateLayer(1, Height:1),
---
>                 .orderOut(nil); self.FramesMovie.WriteMovie(self.Graphics.CreateLayer(1, Height:1),
2775c2808
<       SavePanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in GCUI
---
>       SavePanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in self
2787,2788c2820,2821
<              .Vals("InDatabaseExplorer").VB ? Int(DatabaseGameExplorerSlider.intValue) : GCUI
<              .Attributes.Inf.GetCount()), Size:1320, UpdateBoard:false, Table:false)
```

```
---
>              .Vals("InDatabaseExplorer").VB ? Int(DatabaseGameExplorerSlider.intValue) : Attributes
>              .Inf.GetCount()), Size:1320, UpdateBoard:false, Table:false)
2821c2854
<       DataWritePGN(GameData, Filename, GCUI.Attributes.BlackStarts())
---
>       DataWritePGN(GameData, Filename, Attributes.BlackStarts())
2828,2829c2861
<       (Plies, PGN) = (BoolTo().Num(GCUI.Attributes.BlackStarts()), GCUI.Attributes
<       .GetPGNGameHeader())
---
>       (Plies, PGN) = (BoolTo().Num(Attributes.BlackStarts()), Attributes.GetPGNGameHeader())
2832c2864
<       for Index in 0..<GCUI.Attributes.Inf.GetCount()
---
>       for Index in 0..<Attributes.Inf.GetCount()
2835c2867
<          if (Plies == 1) && GCUI.Attributes.BlackStarts()
---
>          if (Plies == 1) && (Attributes.BlackStarts())
2838c2870
<             PGN += GCUI.Attributes.Inf.Vals("FENStartMoveNumber").VS + "... "
---
>             PGN += Attributes.Inf.Vals("FENStartMoveNumber").VS + "... "
2845c2877
<          if (Plies % 2) == 1 {PGN += "\((Plies - 1) / 2 + GCUI.Attributes.FENStart()). "}
---
>          if (Plies % 2) == 1 {PGN += "\((Plies - 1) / 2 + Attributes.FENStart()). "}
2848c2880
<          PGN += GCUI.Attributes.Inf.Vals(Index).VNS
---
>          PGN += Attributes.Inf.Vals(Index).VNS
2851,2855c2883
<          if !GCUI.Attributes.Inf.VLSIsEmpty(Index)
<          {
<             // append the annotation string //
<             PGN += " " + GCUI.Attributes.Inf.Vals(Index).VLS
<          }
---
>          if !Attributes.Inf.VLSIsEmpty(Index) {PGN += " " + Attributes.Inf.Vals(Index).VLS}
2858c2886
<          PGN += ((Plies % 10) == 0) || (Index == (GCUI.Attributes.Inf.GetCount() - 1)) ? "\n" : " "
---
>          PGN += ((Plies % 10) == 0) || (Index == (Attributes.Inf.GetCount() - 1)) ? "\n" : " "
2862c2890
<       if GCUI.Attributes.NoMoves() {PGN += "*\n"}
---
>       if Attributes.NoMoves() {PGN += "*\n"}
2865c2893
<       PGN += GCUI.Attributes.Inf.Vals("EndResult").VS + "\n"
---
>       PGN += Attributes.Inf.Vals("EndResult").VS + "\n"
2878c2906
<       GCUI.Attributes.StoreGameHeaderData(GameData)
---
>       Attributes.StoreGameHeaderData(GameData)
2884c2912
<       for Index in 0..<GCUI.Attributes.Inf.GetCount()
---
>       for Index in 0..<Attributes.Inf.GetCount()
2887c2915
<          MakeMoveOnBoard(GCUI.Attributes.Inf.Vals(Index).VS)
---
>          MakeMoveOnBoard(Attributes.Inf.Vals(Index).VS)
2893c2921
<             DataStoreMove(GameData, BoardPGNLabel(Board), GCUI.Attributes.Inf.Vals(Index).VLS)
---
>             DataStoreMove(GameData, BoardPGNLabel(Board), Attributes.Inf.Vals(Index).VLS)
2908c2936
<       DataAppendPGN(GameData, Filename, GCUI.Attributes.BlackStarts())
---
>       DataAppendPGN(GameData, Filename, Attributes.BlackStarts())
2932,2933c2960,2961
<          BoardInitializeAttributes(OpeningBoard, GCUI.Attributes.Inf.Vals("FENLayout").VDS, "KQkq",
<                                    "w", "-")
---
>          BoardInitializeAttributes(OpeningBoard, Attributes.Inf.Vals("FENLayout").VDS, "KQkq", "w",
>                                    "-")
2939,2940c2967,2968
<          // loop over the move list for this game in the opening book up to the first 8 moves //
<          for MoveListCountIndex in 0..<min(8, OpeningBookMoveList.count)
---
>          // 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)
3014c3042
<       GCUI.Attributes.Inf.RestoreBackupValues(["FENPref", "TagsPref", "ListsPref"], Tag:"Moves")
---
>       Attributes.Inf.RestoreBackupValues(["FENPref", "TagsPref", "ListsPref"], Tag:"Moves")
3058c3086
<    func SaveBoardLayout() {BPData.SaveLayout(); GCUI.Attributes.Inf.SaveValues(["FENPref"])}
---
>    func SaveBoardLayout() {BPData.SaveLayout(); Attributes.Inf.SaveValues(["FENPref"])}
3067c3095
<       GCUI.Attributes.Inf.RestoreSavedValues(["FENPref"])
---
>       Attributes.Inf.RestoreSavedValues(["FENPref"])
3080c3108
<       GCUI.Attributes.Inf.BackupValues(["FENPref", "TagsPref", "ListsPref"], Tag:"Moves")
---
>       Attributes.Inf.BackupValues(["FENPref", "TagsPref", "ListsPref"], Tag:"Moves")
3136c3164
<             GCUI.Attributes.SetFENDefaults()
---
>             Attributes.SetFENDefaults()
3145c3173
<             GCUI.Attributes.EditingCurrentPosition(Board)
---
>             Attributes.EditingCurrentPosition(Board)
3177c3205
<       GCUI.Attributes.Inf.ClearVS("FENPref", Key:"FENCastlingRights")
---
>       Attributes.Inf.ClearVS("FENPref", Key:"FENCastlingRights")
3183c3211
<          GCUI.Attributes.Inf.AddTo("FENPref", Key:"FENCastlingRights", AddOn:"K")
---
>          Attributes.Inf.AddTo("FENPref", Key:"FENCastlingRights", AddOn:"K")
3190c3218
<          GCUI.Attributes.Inf.AddTo("FENPref", Key:"FENCastlingRights", AddOn:"Q")
---
>          Attributes.Inf.AddTo("FENPref", Key:"FENCastlingRights", AddOn:"Q")
3197c3225
<          GCUI.Attributes.Inf.AddTo("FENPref", Key:"FENCastlingRights", AddOn:"k")
---
>          Attributes.Inf.AddTo("FENPref", Key:"FENCastlingRights", AddOn:"k")
3204c3232
<          GCUI.Attributes.Inf.AddTo("FENPref", Key:"FENCastlingRights", AddOn:"q")
---
>          Attributes.Inf.AddTo("FENPref", Key:"FENCastlingRights", AddOn:"q")
3208c3236
<       if GCUI.Attributes.Inf.VSIsEmpty("FENCastlingRights")
---
>       if Attributes.Inf.VSIsEmpty("FENCastlingRights")
3211c3239
<          GCUI.Attributes.Inf.Update("FENPref", VSItems:["FENCastlingRights":"-"])
---
```

```
>           Attributes.Inf.Update("FENPref", VSItems:["FENCastlingRights":"-"])
3250c3278
<       // refresh all the views //
---
>       // refresh all //
3272c3300
<       ([UTType("com.greenchess.pgn")!, .text], "Append PGN", "Append Current Game to PGN File")
---
>       ([UTType("com.greenchess.pgn")!], "Append PGN", "Append Current Game to PGN File")
3275c3303
<       AppendPanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in GCUI
---
>       AppendPanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in self
3292c3320
<       ([UTType("com.greenchess.pgn")!, .text], "Open PGN", "Open PGN Game File")
---
>       ([UTType("com.greenchess.pgn")!], "Open PGN", "Open PGN Game File")
3295c3323
<       OpenPanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in GCUI
---
>       OpenPanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in self
3312,3313c3340,3341
<       SavePanel.message) = ([UTType("com.greenchess.pgn")!, .text], "GreenChessGame.pgn",
<       "Save PGN", "Save Current Game to PGN File")
---
>       SavePanel.message) = ([UTType("com.greenchess.pgn")!], "GreenChessGame.pgn", "Save PGN",
>       "Save Current Game to PGN File")
3316c3344
<       SavePanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in GCUI
---
>       SavePanel.beginSheetModal(for:MainWindow, completionHandler:{Result -> Void in self
3326c3354
<       GCUI.Attributes.Inf.RestoreBackupValues(["FENPref", "TagsPref", "ListsPref"], Tag:"Moves")
---
>       Attributes.Inf.RestoreBackupValues(["FENPref", "TagsPref", "ListsPref"], Tag:"Moves")
3328c3356
<       // refresh all the views //
---
>       // refresh all //
3394c3422
<       GCUI.Attributes.SaveSubset(Int(GameExplorerSlider.intValue))
---
>       Attributes.SaveSubset(Int(GameExplorerSlider.intValue))
3396c3424
<       // refresh all the views //
---
>       // refresh all //
3413c3441
<       // refresh all the views //
---
>       // refresh all //
3426c3454
<       // refresh all the views //
---
>       // refresh all //
3437c3465
<       GCUI.Attributes.ProcessGameData(GameData)
---
>       Attributes.ProcessGameData(GameData)
3482c3510
<    func ExitPreferencesWindow()
---
>    @IBAction func ExitPreferencesWindow(_:AnyObject)
3500c3528
<       GCUI.BPComp.SavePreferences()
---
>       BPComp.SavePreferences()
3503c3531
<       GCUI.FramesMovie.SavePrefs()
---
>       FramesMovie.SavePrefs()
3518c3546
<       ProgressCircle.UpdateCircleColor(GCUI.GetColorView("WindowColorView").GetCurrentHue())
---
>       ProgressCircle.UpdateCircleColor(Ctr.WindowBackgroundColorView.GetCurrentHue())
3534c3562
<       GCUI.Attributes.UpdateFromEnterGameInfoWindow()
---
>       Attributes.UpdateFromEnterGameInfoWindow()
3536c3564
<       // refesh all the views //
---
>       // refesh the views //
3574c3602
<       UpdateSlider(GameGallerySlider, MaxV:GCUI.FramesMovie.MaxPosition, IntV:0)
---
>       UpdateSlider(GameGallerySlider, MaxV:GameGalleryView.FramesMovie.MaxPosition, IntV:0)
3595,3599c3623
<    func UpdateGameListFont(_ Key:String, Selection:Int)
<    {
<       // update the game list font //
<       GameListFont.UpdateFont(0, UseVertOffset:false, Key:Key, Selection:Selection)
<    }
---
>    @IBAction func UpdateGameListFont(_:AnyObject) {GameListFont.UpdateFont(0, UseVertOffset:false)}
3602,3610c3626
<    func UpdateMoveListFont(_ Key:String, Selection:Int)
<    {
<       // update the move list font //
<       MoveListFont.UpdateFont(Inf.Vals("IllustrationSize").VI, UseVertOffset:!Inf
<                             .Vals("HideIllustrations").VB, Key:Key, Selection:Selection)
<
<       // refresh the tables //
<       Refresh("Tables")
<    }
---
>    @IBAction func UpdateMoveListFont(_:AnyObject) {Refresh("Tables")}
3613,3620c3629
<    func UpdateGameInfoFont(_ Key:String, Selection:Int)
<    {
<       // update the game info font //
<       GameInfoFont.UpdateFont(0, UseVertOffset:false, Key:Key, Selection:Selection)
<
<       // refresh the tables //
<       Refresh("Tables")
<    }
---
>    @IBAction func UpdateGameInfoFont(_:AnyObject) {Refresh("Tables")}
3623c3632
<    func UpdateBoardLabelsFont(_ Key:String, Selection:Int)
---
>    @IBAction func UpdateBoardLabelsFont(_:AnyObject)
3626c3635
<       GCUI.BPComp.BoardLabelsFont.UpdateFont(0, UseVertOffset:false, Key:Key, Selection:Selection)
---
>       BPComp.BoardLabelsFont.UpdateFont(0, UseVertOffset:false)
3636,3637c3645,3646
<       Inf.Update("VIPref", VIItems:["GameClockBackgroundHue":GCUI
<                 .GetColorView("ClockSurfaceColorView").GetHueValue()])
---
>       Inf.Update("VIPref", VIItems:["GameClockBackgroundHue":Ctr.ClockBackgroundColorView
>                 .GetHueValue()])
3640,3641c3649
<       WhiteTimeClock.UpdateClockBackgroundColor(GCUI.GetColorView("ClockSurfaceColorView")
<                                                  .GetCurrentHue())
---
>       WhiteTimeClock.UpdateClockBackgroundColor(Ctr.ClockBackgroundColorView.GetCurrentHue())
3644,3645c3652
<       BlackTimeClock.UpdateClockBackgroundColor(GCUI.GetColorView("ClockSurfaceColorView")
<                                                  .GetCurrentHue())
---
>       BlackTimeClock.UpdateClockBackgroundColor(Ctr.ClockBackgroundColorView.GetCurrentHue())
3648,3649c3655
<       GCUI.GetClockView("ClockView").UpdateClockBackgroundColor(GCUI
<                                        .GetColorView("ClockSurfaceColorView").GetCurrentHue())
---
>       Ctr.TimeClockPreview.UpdateClockBackgroundColor(Ctr.ClockBackgroundColorView.GetCurrentHue())
3656,3657c3662
<       Inf.Update("VIPref", VIItems:["GameClockHue":GCUI.GetColorView("ClockColorView")
<                 .GetHueValue()])
---
>       Inf.Update("VIPref", VIItems:["GameClockHue":Ctr.ClockColorView.GetHueValue()])
3660c3665
<       WhiteTimeClock.UpdateClockBorderColor(GCUI.GetColorView("ClockColorView").GetCurrentHue())
---
>       WhiteTimeClock.UpdateClockBorderColor(Ctr.ClockColorView.GetCurrentHue())
3663c3668
<       BlackTimeClock.UpdateClockBorderColor(GCUI.GetColorView("ClockColorView").GetCurrentHue())
---
>       BlackTimeClock.UpdateClockBorderColor(Ctr.ClockColorView.GetCurrentHue())
3666,3667c3671
<       GCUI.GetClockView("ClockView").UpdateClockBorderColor(GCUI.GetColorView("ClockColorView")
<                                                               .GetCurrentHue())
---
>        Ctr.TimeClockPreview.UpdateClockBorderColor(Ctr.ClockColorView.GetCurrentHue())
3671c3675
<     func SetNumberOfGridRows(_ Key:String, Selection:Int)
---
>     @IBAction func SetNumberOfGridRows(_:AnyObject)
3674c3678
<        Inf.Update("VIPref", VIItems:[Key:Selection + 2])
---
>        Inf.Update("VIPref", VIItems:["NumberOfGridRows":Ctr.GridRowsButton.indexOfSelectedItem + 2])
3678c3682
<     func PreferencesAction(_ Value:Int)
---
>     @IBAction func PreferencesAction(_:AnyObject)
3680,3682c3684,3687
<        // exit the preferences window is click on close button //
<        if Value == 0 {ExitPreferencesWindow(); return}
<
---
>        // get the current preferences selection //
>        Inf.Update("VIPref", VIItems:["PreferencesSelection":Int(PreferencesToolbar
>                 .selectedItemIdentifier!.rawValue)!])
>
3684,3686c3689,3695
<        Inf.Update("VIPref", VIItems:["PreferencesSelection":Value])
<
<        // update the board previews //
---
>        UpdateCurrentPreferencesSelection()
>     }
>
>     // update the current preferences selections //
>     func UpdateCurrentPreferencesSelection()
>     {
>        // update the board and diagram previews //
3688,3690c3697,3730
<
<        // update the visible toolbar selection buttons //
<        GCUI.UpdateVisibility(GCUI.PrefsBar[Value].3)
---
>
>        // return if the preferences selection is the same as the previous //
>        if Inf.VIIsEqual("PreferencesSelection", VI:PreviousPreferencesSelection) {return}
>
>        // if the previously selected preference controls exist //
>        if Ctr.PrefControls[PreviousPreferencesSelection] != nil
>        {
>           // hide the previous preferences selection //
>           Ctr.PrefControls[PreviousPreferencesSelection]!.UpdateState(true)
>        }
>
>        // if the previously selected preference controls exist //
>        if PrefsSubWindows[PreviousPreferencesSelection] != nil
>        {
>           // hide the previous preferences selection //
>           UI.ObjsVisible(false, SubWindow:PrefsSubWindows[PreviousPreferencesSelection]!)
>        }
>
>        // if the preferences controls exist //
>        if Ctr.PrefControls[Inf.Vals("PreferencesSelection").VI] != nil
>        {
>           // display the current preferences selection //
>           Ctr.PrefControls[Inf.Vals("PreferencesSelection").VI]!.UpdateState(false)
>        }
>
>        // if the preference controls exist //
>        if PrefsSubWindows[Inf.Vals("PreferencesSelection").VI] != nil
>        {
>           // display the current preferences selection //
>           UI.ObjsVisible(true, SubWindow:PrefsSubWindows[Inf.Vals("PreferencesSelection").VI]!)
>        }
>
>        // save the previous preferences selection //
>        PreviousPreferencesSelection = Inf.Vals("PreferencesSelection").VI
3700c3740
<        PreferencesAction(Inf.Vals("PreferencesSelection").VI)
---
>        UpdateCurrentPreferencesSelection()
3980c4020
<        for TheClock in [WhiteTimeClock, BlackTimeClock, GCUI.GetClockView("ClockView")]
---
>        for TheClock in [WhiteTimeClock, BlackTimeClock, Ctr.TimeClockPreview]
4029c4069
<        UpdateSlider(GameAnnotatorSlider, MaxV:GCUI.Attributes.Inf.GetCount() - 1, IntV:0)
---
>        UpdateSlider(GameAnnotatorSlider, MaxV:Attributes.Inf.GetCount() - 1, IntV:0)
4035c4075
<        if GCUI.Attributes.NoMoves()
---
>        if Attributes.NoMoves()
4084,4085c4124
<        GCUI.Attributes.ExtractEnteredFENString(GetString().FromTextField(EnterFENTextField
<                                                 .stringValue))
---
>        Attributes.ExtractEnteredFENString(GetString().FromTextField(EnterFENTextField.stringValue))
4120c4159
<        if !GCUI.Attributes.NoMoves()
---
>        if !Attributes.NoMoves()
4129c4168
<        // refresh all the views //
---
>        // refresh all //
4218,4219c4257,4258
<        UpdateSlider(GameExplorerSlider, MaxV:GCUI.Attributes.Inf.GetCount(), IntV:
<                     Int(GameExplorerSlider.intValue))
---
>        UpdateSlider(GameExplorerSlider, MaxV:Attributes.Inf.GetCount(), IntV:Int(GameExplorerSlider
>                     .intValue))
4271c4310
<        if GCUI.Attributes.NoMoves() {return}
---
>        if Attributes.NoMoves() {return}
4310,4311c4349,4350
<        UpdateSlider(DatabaseGameExplorerSlider, MaxV:GCUI.Attributes.Inf.GetCount(), IntV:GCUI
<                     .Attributes.Inf.GetCount())
---
>        UpdateSlider(DatabaseGameExplorerSlider, MaxV:Attributes.Inf.GetCount(), IntV:Attributes.Inf
>                     .GetCount())
4335,4336c4374
<        GCUI.Attributes.Inf.Update("Moves", VLSItems:[String(GameAnnotatorMoveLocation):
<                                   AnnotationString])
---
>        Attributes.Inf.Update("Moves", VLSItems:[String(GameAnnotatorMoveLocation):AnnotationString])
4343c4381
<        if GCUI.Attributes.NoMoves() {return}
---
>        if Attributes.NoMoves() {return}
4346c4384
<        AnnotationString = GCUI.Attributes.Inf.Vals(MoveLocation).VLS
---
>        AnnotationString = Attributes.Inf.Vals(MoveLocation).VLS
4367c4405
<        if MoveLocation + 2 <= GCUI.Attributes.Inf.GetCount()
```

```
---
>       if MoveLocation + 2 <= Attributes.Inf.GetCount()
4421c4459
<     func UpdateClockBackgroundVisibility(_ Key:String, Value:Double) -> String
---
>     func UpdateClockBackgroundVisibility(_ Value:Double) -> String
4424c4462
<       Inf.Update("VIPref", VIItems:[Key:Int(Value)])
---
>       Inf.Update("VIPref", VIItems:["ClockBackgroundVisibility":Int(Value)])
4430c4468
<       return "Clock background" + Inf.Perc(Key)
---
>       return "Clock background" + Inf.Perc("ClockBackgroundVisibility")
4437c4475
<       for TheClock in [WhiteTimeClock, BlackTimeClock, GCUI.GetClockView("ClockView")]
---
>       for TheClock in [WhiteTimeClock, BlackTimeClock, Ctr.TimeClockPreview]
4445c4483
<     func UpdateWindowTextureOpacity(_ Key:String, Value:Double) -> String
---
>     func UpdateWindowTextureOpacity(_ Value:Double) -> String
4448c4486
<       Inf.Update("VIPref", VIItems:[Key:Int(Value)])
---
>       Inf.Update("VIPref", VIItems:["WindowTextureOpacity":Int(Value)])
4454c4492
<       return "Window texture" + Inf.Perc(Key)
---
>       return "Window texture" + Inf.Perc("WindowTextureOpacity")
4458c4496
<     func UpdateAnimationRate(_ Key:String, Value:Double) -> String
---
>     func UpdateAnimationRate(_ Value:Double) -> String
4461c4499
<       Inf.Update("VIPref", VIItems:[Key:Int(Value)])
---
>       Inf.Update("VIPref", VIItems:["AnimationRate":Int(Value)])
4464c4502
<       return "Board animation speed is " + Inf.IntStr(Key)
---
>       return "Board animation speed is " + Inf.IntStr("AnimationRate")
4468c4506
<     func UpdateGalleryAnimationRate(_ Key:String, Value:Double) -> String
---
>     func UpdateGalleryAnimationRate(_ Value:Double) -> String
4471c4509
<       Inf.Update("VIPref", VIItems:[Key:Int(Value)])
---
>       Inf.Update("VIPref", VIItems:["GalleryAnimationRate":Int(Value)])
4474c4512
<       return "Gallery animation speed is " + Inf.IntStr(Key)
---
>       return "Gallery animation speed is " + Inf.IntStr("GalleryAnimationRate")
4478c4516
<     func UpdateIllustrationSize(_ Key:String, Value:Double) -> String
---
>     func UpdateIllustrationSize(_ Value:Double) -> String
4481c4519
<       Inf.Update("VIPref", VIItems:[Key:Int(Value)])
---
>       Inf.Update("VIPref", VIItems:["IllustrationSize":Int(Value)])
4487c4525
<       return "Diagram size is " + Inf.IntStr(Key)
---
>       return "Diagram size is " + Inf.IntStr("IllustrationSize")
4491c4529
<     func UpdateUCIEngineDifficulty(_ Key:String, Value:Double) -> String
---
>     func UpdateUCIEngineDifficulty(_ Value:Double) -> String
4494c4532
<       Inf.Update("VIPref", VIItems:[Key:Int(Value)])
---
>       Inf.Update("VIPref", VIItems:["UCIEngineDifficulty":Int(Value)])
4497c4535
<       return "Stockfish Engine Difficulty is " + Inf.IntStr(Key)
---
>       return "Difficulty is " + Inf.IntStr("UCIEngineDifficulty")
4501c4539
<     func UpdateUCIEngineHash(_ Key:String, Value:Double) -> String
---
>     func UpdateUCIEngineHash(_ Value:Double) -> String
4504c4542
<       Inf.Update("VIPref", VIItems:[Key:Int(Value)])
---
>       Inf.Update("VIPref", VIItems:["UCIEngineHash":Int(Value)])
4507c4545
<       return "Stockfish Engine Hash is " + Inf.IntStr(Key) + " MB"
---
>       return "Hash is " + Inf.IntStr("UCIEngineHash") + " MB"
4511c4549
<     func UpdateSearchTime(_ Key:String, Value:Double) -> String
---
>     func UpdateSearchTime(_ Value:Double) -> String
4514c4552
<       Inf.Update("VIPref", VIItems:[Key:Int(Value)])
---
>       Inf.Update("VIPref", VIItems:["SearchTime":Int(Value)])
4517c4555
<       return "Engine search time is " + Inf.IntStr(Key) + " seconds"
---
>       return "Engine search time is " + Inf.IntStr("SearchTime") + " seconds"
4556c4594
<       BPData.UpdateStartAndEndSquareForMove(MovingPiece.UCIMove, Dims:Dims, Rotated:GCUI.BPComp.Inf
---
>       BPData.UpdateStartAndEndSquareForMove(MovingPiece.UCIMove, Dims:Dims, Rotated:BPComp.Inf
4562,4569c4600
< 
<     // get the updated control state //
<     func UpdatedState(_ State:Bool) -> NSControl.StateValue
<     {
<       // return the updated control state //
<       return NSControl.StateValue(rawValue:BoolTo().Num(State))
<     }
< 
---
> 
4583,4584c4614,4615
<       (UpdatedState(Inf.Vals("HumanOpponent").VB), UpdatedState(Inf.Vals("EngineOpponent")
<       .VB), UpdatedState(Inf.Vals("UCIEngineOpponent").VB))
---
>       (Ctr.UpdatedState(Inf.Vals("HumanOpponent").VB), Ctr.UpdatedState(Inf.Vals("EngineOpponent")
>       .VB), Ctr.UpdatedState(Inf.Vals("UCIEngineOpponent").VB))
4597c4628
<       PGNLabel = GCUI.Attributes.Inf.Vals(Index).VNS
---
>       PGNLabel = Attributes.Inf.Vals(Index).VNS
4603c4634
<       if ((Index + GCUI.Attributes.MoveOffset()) % 2) == 0
---
>       if ((Index + Attributes.MoveOffset()) % 2) == 0
4630,4631c4661
<       PGNMove = String(format:"%d.", (Index + GCUI.Attributes.MoveOffset()) / 2 + GCUI.Attributes
<                     .FENStart())
```

```
---
>       PGNMove = String(format:"%d.", (Index + Attributes.MoveOffset()) / 2 + Attributes.FENStart())
4634c4664
<       if ((Index + GCUI.Attributes.MoveOffset()) % 2) == 1 {PGNMove += ".."}
---
>       if ((Index + Attributes.MoveOffset()) % 2) == 1 {PGNMove += ".."}
4645c4675
<                               .Vals("HideIllustrations").VB, Key:"", Selection:0)
---
>                               .Vals("HideIllustrations").VB)
4648c4678
<       GameInfoFont.UpdateFont(0, UseVertOffset:false, Key:"", Selection:0)
---
>       GameInfoFont.UpdateFont(0, UseVertOffset:false)
4666,4667c4696,4697
<           GameInfoList.append(String(format:"Move %d", (GCUI.Attributes.Inf.GetCount() + GCUI
<                           .Attributes.MoveOffset()) / 2 + GCUI.Attributes.FENStart()))
---
>           GameInfoList.append(String(format:"Move %d", (Attributes.Inf.GetCount() + Attributes
>                           .MoveOffset()) / 2 + Attributes.FENStart()))
4680c4710
<       GameInfoList += GCUI.Attributes.GetInfoList()
---
>       GameInfoList += Attributes.GetInfoList()
4686c4716
<           GCUI.Attributes.ProcessAnalysis(StockfishPV);
---
>           Attributes.ProcessAnalysis(StockfishPV);
4696c4726
<       for Index in 0..<GCUI.Attributes.Inf.AnalysisCount()
---
>       for Index in 0..<Attributes.Inf.AnalysisCount()
4702c4732
<           if Index < GCUI.Attributes.Inf.GetCount()
---
>           if Index < Attributes.Inf.GetCount()
4705c4735
<               AnnotationString = GCUI.Attributes.Inf.Vals(Index).VLS
---
>               AnnotationString = Attributes.Inf.Vals(Index).VLS
4740c4770
<       GCUI.PGNMovesData.RefreshPGNMovesList(PGNMovesList, MoveListFont:MoveListFont)
---
>       PGNMovesData.RefreshPGNMovesList(PGNMovesList, MoveListFont:MoveListFont)
4752c4782
<       PGNMovesTableView.scrollRowToVisible(GCUI.PGNMovesData.LastRow())
---
>       PGNMovesTableView.scrollRowToVisible(PGNMovesData.LastRow())
4785c4815
<       BPData.UpdateStartAndEndSquareForMove(MovingPiece.UCIMove, Dims:Dims, Rotated:GCUI.BPComp.Inf
---
>       BPData.UpdateStartAndEndSquareForMove(MovingPiece.UCIMove, Dims:Dims, Rotated:BPComp.Inf
4799c4829
<       GCUI.GameSounds.Play("click")
---
>       GameSounds.Play("click")
4802c4832
<       GCUI.Attributes.SwapColorToMove()
---
>       Attributes.SwapColorToMove()
4814c4844
<       EditorHotspotPoint) : NSCursor.init(image:GCUI.BPComp.GetPieceNSImage(Type), hotSpot:
---
>       EditorHotspotPoint) : NSCursor.init(image:BPComp.GetPieceNSImage(Type), hotSpot:
4819c4849
<   @IBAction func PositionEditorEraser(_:AnyObject) {GCUI.GameSounds.Play("click"); SetCursor(0)}
---
>   @IBAction func PositionEditorEraser(_:AnyObject) {GameSounds.Play("click"); SetCursor(0)}
4822c4852
<   @IBAction func PositionEditorEPAction(_:AnyObject) {GCUI.GameSounds.Play("click"); SetCursor(10)}
---
>   @IBAction func PositionEditorEPAction(_:AnyObject) {GameSounds.Play("click"); SetCursor(10)}
4853c4883
<       GCUI.GameSounds.Play("click")
---
>       GameSounds.Play("click")
4866c4896
<       GCUI.GameSounds.Play("click")
---
>       GameSounds.Play("click")
4873c4903
<       GCUI.GameSounds.Play("click")
---
>       GameSounds.Play("click")
4887a4918,4920
>
>   // toggle the preference //
>   func TogglePref(_ Key:String) {GameSounds.Play("click"); Inf.ToggleVB("VBPref", Key:Key)}
4889,4890c4922,4923
<   // update the multithread stockfish engine option //
<   func UpdateMultithreadStockfish(_ Key:String) {Inf.ToggleVB("VBPref", Key:Key)}
---
>   // the multithread uci engine button action //
>   @IBAction func MultithreadUCIEngineButton(_:AnyObject) {TogglePref("MultithreadUCIEngine")}
4892,4893c4925,4926
<   // update autosave game toggle button //
<   func UpdateAutosaveGame(_ Key:String) {Inf.ToggleVB("VBPref", Key:Key); StoreAutosaveData("")}
---
>   // the autosave game button action //
>   @IBAction func AutosaveGame(_:AnyObject) {TogglePref("AutosaveGame"); StoreAutosaveData("")}
4895,4899c4928,4929
<   // update if using the opening book //
<   func UpdateUseEcoBook(_ Key:String) {Inf.ToggleVB("VBPref", Key:Key)}
<
<   // update the hide illustrations in game moves option //
<   func UpdateHideIllustrations(_ Key:String) {Inf.ToggleVB("VBPref", Key:Key); Refresh("All")}
---
>   // activate or deactivate using the encyclopedia of chess openings //
>   @IBAction func EcoBookButtonAction(_:AnyObject) {TogglePref("UseEcoBook")}
4900a4931,4933
>   // display the illustrations action //
>   @IBAction func DisplayIllus(_:AnyObject) {TogglePref("HideIllustrations"); Refresh("All")}
>
4960c4993
<       if GCUI.FramesMovie.MaxPosition == 0 {return}
---
>       if GameGalleryView.FramesMovie.MaxPosition == 0 {return}
5046,5048c5079,5081
<       (WhiteMovesToolbarButton.image, WhiteMovesToolbarButton.label) = (GCUI.BPComp
<       .GetPieceNSImage(GCUI.Attributes.WhiteStarts() ? 1 : -1), (GCUI.Attributes.WhiteStarts() ?
<       "White" : "Black") + " Moves")
---
>       (WhiteMovesToolbarButton.image, WhiteMovesToolbarButton.label) = (BPComp
>       .GetPieceNSImage(Attributes.WhiteStarts() ? 1 : -1), (Attributes.WhiteStarts() ? "White" :
>       "Black") + " Moves")
5054,5055c5087
<           TheItem.image =  GCUI.BPComp.GetPieceNSImage(PositionPieces[TheItem.itemIdentifier
<                                                   .rawValue]!)
---
>           TheItem.image = BPComp.GetPieceNSImage(PositionPieces[TheItem.itemIdentifier.rawValue]!)
5088c5120
<       GCUI.GameSounds.Play("buttonclick")
```

```
---
>       GameSounds.Play("buttonclick")
5252c5284
<       if !GCUI.Attributes.Inf.UndoToPreviousVS() {return}
---
>       if !Attributes.Inf.UndoToPreviousVS() {return}
5254c5286
<       // refresh all the views //
---
>       // refresh all //
5271c5303
<       for Index in 0..<GCUI.Attributes.Inf.AnalysisCount()
---
>       for Index in 0..<Attributes.Inf.AnalysisCount()
5274c5306
<          GCUI.Attributes.Inf.MarkValid("Moves", Key:Index)
---
>          Attributes.Inf.MarkValid("Moves", Key:Index)
5277c5309
<          MakeMoveOnBoard(GCUI.Attributes.Inf.Vals(Index).VS)
---
>          MakeMoveOnBoard(Attributes.Inf.Vals(Index).VS)
5283,5284c5315,5316
<             GCUI.Attributes.Inf.Update("Moves", VNSItems:[Index:String(validatingUTF8:
<                                        BoardPGNLabel(Board))!])
---
>             Attributes.Inf.Update("Moves", VNSItems:[Index:String(validatingUTF8:
>                                   BoardPGNLabel(Board))!])
5290c5322
<             GCUI.Attributes.Inf.MarkInvalid("Moves", Key:Index)
---
>             Attributes.Inf.MarkInvalid("Moves", Key:Index)
5295c5327
<       GCUI.Attributes.Compress(Type)
---
>       Attributes.Compress(Type)
5301,5305c5333
<       for Index in 0..<GCUI.Attributes.Inf.GetCount()
<       {
<          // make the move on the board //
<          MakeMoveOnBoard(GCUI.Attributes.Inf.Vals(Index).VS)
<       }
---
>       for Index in 0..<Attributes.Inf.GetCount() {MakeMoveOnBoard(Attributes.Inf.Vals(Index).VS)}
5315c5343
<       for Index in 0..<GCUI.Attributes.Inf.GetCount()
---
>       for Index in 0..<Attributes.Inf.GetCount()
5318c5346
<          MovingPiece.SaveUCIMove(GCUI.Attributes.Inf.Vals(Index).VS)
---
>          MovingPiece.SaveUCIMove(Attributes.Inf.Vals(Index).VS)
5349,5353c5377,5381
<       return GCUI.BPComp.Recomposite(Size, BPData:((GCUI.Attributes.Inf.GetCount() == Index) &&
<              !Table) ? BPData : GCUI.PGNMovesData.GetData(Index), InPositionEditor:Inf
<              .Vals("InPositionEditor").VB, FENEnpassantSquare:GCUI.Attributes.Inf
<              .Vals("FENEnpassantSquare").VS, EnpassantSquare:EnpassantSquare, RankAndFile:
<              MovingPiece.Location, UpdateBoard:UpdateBoard, Info:false)
---
>       return BPComp.Recomposite(Size, BPData:((Attributes.Inf.GetCount() == Index) && !Table) ?
>                    BPData : PGNMovesData.GetData(Index), InPositionEditor:Inf
>                    .Vals("InPositionEditor").VB, FENEnpassantSquare:Attributes.Inf
>                    .Vals("FENEnpassantSquare").VS, EnpassantSquare:EnpassantSquare,
>                    RankAndFile:MovingPiece.Location, UpdateBoard:UpdateBoard, Info:false)
5360c5388
<       return FrameCGImage(GCUI.Attributes.Inf.GetCount(), Size:Size, UpdateBoard:false, Table:false)
---
>       return FrameCGImage(self.Attributes.Inf.GetCount(), Size:Size, UpdateBoard:false, Table:false)
5370c5398
<       if Board {GCUI.Objs["BoardPreview"]!.Preview = GetPreview(800)}
---
>       if Board {UI.Objs["BoardPreview"]!.Preview = GetPreview(800)}
5373c5401
<       if Diagram {GCUI.Objs["DiagramPreview"]!.Preview = GetPreview(Inf.VF("IllustrationSize") * 2)}
---
>       if Diagram {UI.Objs["DiagramPreview"]!.Preview = GetPreview(Inf.VF("IllustrationSize") * 2)}
5376c5404
<       if Texture {GCUI.Objs["WindowPreview"]!.Preview = WindowTexture.Preview}
---
>       if Texture {UI.Objs["WindowPreview"]!.Preview = WindowTexture.Preview}
5383c5411
<       if OnMain {DispatchQueue.main.async
---
>       if OnMain
5385,5387c5413,5419
<          // update the previews //
<          self.UpdatePreviews(RefreshBoard, Board:Board, Diagram:Diagram, Texture:Texture)
<       }}
---
>          // update the board preview images on the main thread //
>          DispatchQueue.main.async
>          {
>             // update the previews //
>             self.UpdatePreviews(RefreshBoard, Board:Board, Diagram:Diagram, Texture:Texture)
>          }
>       }
5406c5438
<    func UpdateIllustrationData(_ Index:Int) {GCUI.PGNMovesData.Update(Index, BPData:BPData)}
---
>    func UpdateIllustrationData(_ Index:Int) {PGNMovesData.Update(Index, BPData:BPData)}
5428,5429c5460,5461
<       MainViewImage = FrameCGImage(GCUI.Attributes.Inf.GetCount(), Size:1320, UpdateBoard:
<                                    UpdateBoard, Table:false)
---
>       MainViewImage = FrameCGImage(Attributes.Inf.GetCount(), Size:1320, UpdateBoard:UpdateBoard,
>                                    Table:false)
```