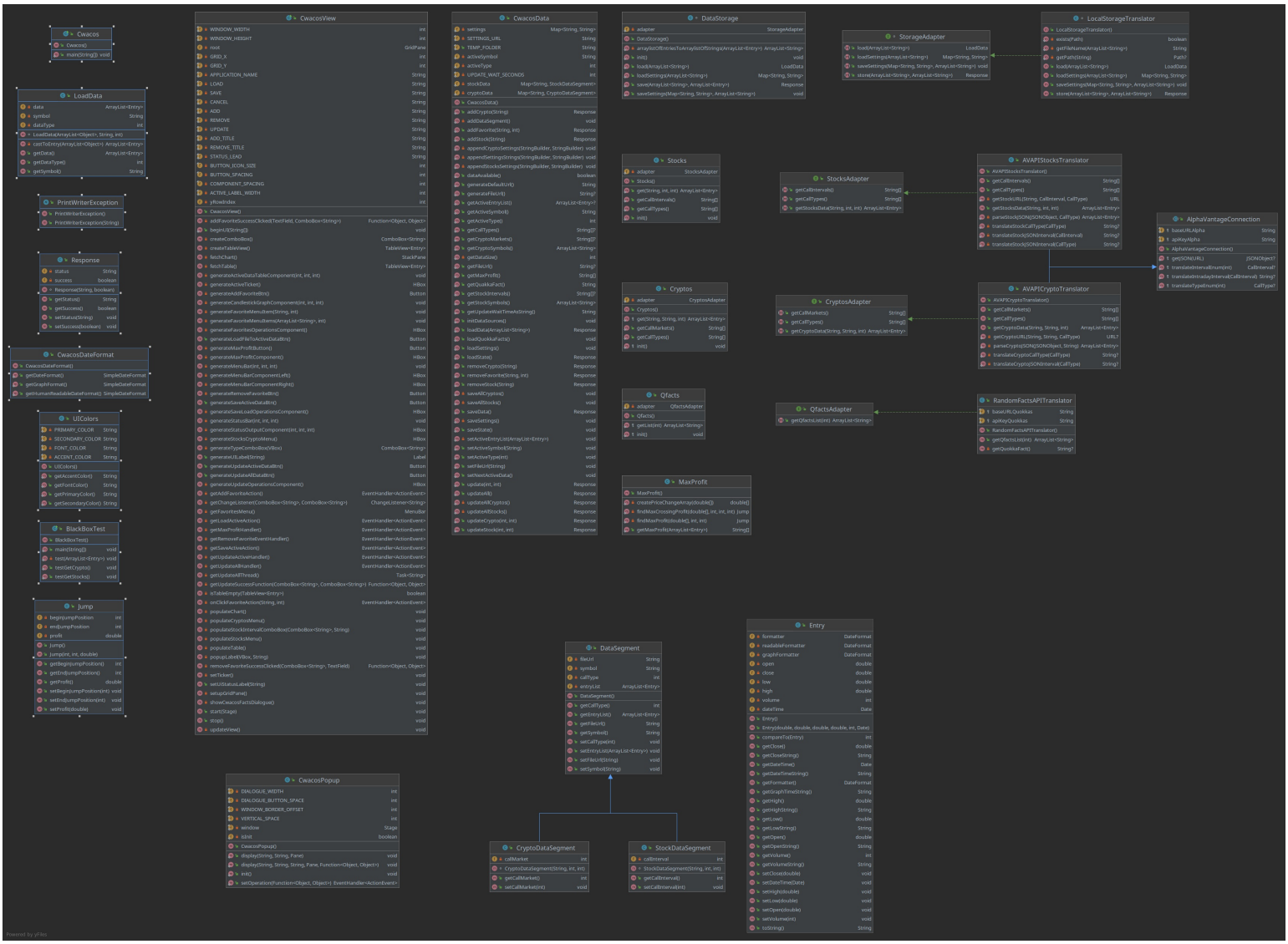
Cwacos
Cwacos()
main(String[]) void
LoadData
data ArrayList<Entry>
symbol String
dataType int
LoadData(ArrayList<Object>, String, int)
castToEntry(ArrayList<Object>) ArrayList<Entry>
getData() ArrayList<Entry>
getDataType() int
getSymbol() String
PrintWriterException
PrintWriterException()
PrintWriterException(String)
Response
status String
success boolean
Response(String, boolean)
getStatus() String
getSuccess() boolean
setStatus(String) void
setSuccess(boolean) void
CwacosDateFormat
CwacosDateFormat()
getDateFormat() SimpleDateFormat
getGraphFormat() SimpleDateFormat
getHumanReadableDateFormat() SimpleDateFormat
UIColors
PRIMARY_COLOR String
SECONDARY_COLOR String
FONT_COLOR String
ACCENT_COLOR String
UIColors()
getAccentColor() String
getFontColor() String
getPrimaryColor() String
getSecondaryColor() String
BlackBoxTest
BlackBoxTest()
main(String[]) void
test(ArrayList<Entry>) void
testGetCrypto() void
testGetStocks() void
Jump
beginJumpPosition int
endJumpPosition int
profit double
Jump()
Jump(int, int, double)
getBeginJumpPosition() int
getEndJumpPosition() int
getProfit() double
setBeginJumpPosition(int) void
setEndJumpPosition(int) void
setProfit(double) void
CwacosView
WINDOW_WIDTH int
WINDOW_HEIGHT int
root GridPane
GRID_X int
GRID_Y int
APPLICATION_NAME String
LOAD String
SAVE String
CANCEL String
ADD String
REMOVE String
UPDATE String
ADD_TITLE String
REMOVE_TITLE String
STATUS_LEAD String
BUTTON_ICON_SIZE int
BUTTON_SPACING int
COMPONENT_SPACING int
ACTIVE_LABEL_WIDTH int
yRowIndex int
CwacosView()
addFavoriteSuccessClicked(TextField, ComboBox<String>) Function<Object, Object>
beginUI(String[]) void
createComboBox() ComboBox<String>
createTableView() TableView<Entry>
fetchChart() StackPane
fetchTable() TableView<Entry>
generateActiveDataTableComponent(int, int, int) void
generateActiveTicker() HBox
generateAddFavoriteBtn() Button
generateCandlestickGraphComponent(int, int, int) void
generateFavoriteMenuItem(String, int) void
generateFavoriteMenuItems(ArrayList<String>, int) void
generateFavoritesOperationsComponent() HBox
generateLoadFileToActiveDataBtn() Button
generateMaxProfitButton() Button
generateMaxProfitComponent() HBox
generateMenuBar(int, int, int) void
generateMenuBarComponentLeft() HBox
generateMenuBarComponentRight() HBox
generateRemoveFavoriteBtn() Button
generateSaveActiveDataBtn() Button
generateSaveLoadOperationsComponent() HBox
generateStatusBar(int, int, int) void
generateStatusOutputComponent(int, int, int) HBox
generateStocksCryptoMenu() HBox
generateTypeComboBox() ComboBox<String>
generateUILabel(String) Label
generateUpdateActiveDataBtn() Button
generateUpdateAllDataBtn() Button
generateUpdateOperationsComponent() HBox
getAddFavoriteAction() EventHandler<ActionEvent>
getChangeListener(ComboBox<String>, ComboBox<String>) ChangeListener<String>
getFavoritesMenu() MenuBar
getLoadActiveAction() EventHandler<ActionEvent>
getMaxProfitHandler() EventHandler<ActionEvent>
getRemoveFavoriteEventHandler() EventHandler<ActionEvent>
getSaveActiveAction() EventHandler<ActionEvent>
getUpdateActiveHandler() EventHandler<ActionEvent>
getUpdateAllHandler() EventHandler<ActionEvent>
getUpdateAllThread() Task<String>
getUpdateSuccessFunction(ComboBox<String>, ComboBox<String>) Function<Object, Object>
isTableEmpty(TableView<Entry>) boolean
onClickFavoriteAction(String, int) EventHandler<ActionEvent>
populateChart() void
populateCryptoMenu() void
populateStockIntervalComboBox(ComboBox<String>, String) void
populateStocksMenu() void
populateTable() void
popupLabelVBox, String) void
removeFavoriteSuccessClicked(ComboBox<String>, TextField) Function<Object, Object>
setTicker() void
setUIStatusLabel(String) void
setupGridPane() void
showCwacosFactsDialogue() void
start(Stage) void
stop() void
updateView() void
CwacosPopup
DIALOGUE_WIDTH int
DIALOGUE_BUTTON_SPACE int
WINDOW_BORDER_OFFSET int
VERTICAL_SPACE int
window Stage
isInit boolean
CwacosPopup()
display(String, String, Pane) void
display(String, String, String, Pane, Function<Object, Object>) void
init() void
setOperation(Function<Object, Object>) EventHandler<ActionEvent>
CwacosData
settings Map<String, String>
SETTINGS_URL String
TEMP_FOLDER String
activeSymbol String
activeType int
UPDATE_WAIT_SECONDS int
stockData Map<String, StockDataSegment>
cryptoData Map<String, CryptoDataSegment>
CwacosData()
addCrypto(String) Response
addDataSegment() void
addFavorite(String, int) Response
addStock(String) Response
appendCryptoSettings(StringBuilder, StringBuilder) void
appendSettingsStrings(StringBuilder, StringBuilder) void
appendStocksSettings(StringBuilder, StringBuilder) void
dataAvailable() boolean
generateDefaultUrl() String
generateFileUrl() String?
getActiveEntryList() ArrayList<Entry>?
getActiveSymbol() String
getActiveType() int
getCallTypes() String[]?
getCryptoMarkets() String[]?
getCryptoSymbols() ArrayList<String>
getDataSize() int
getFileUrl() String?
getMaxProfit() String[]
getQuokkaFact() String
getStockInterval() String[]?
getStockSymbols() ArrayList<String>
getUpdateWaitTimeAsString() String
initDataSources() void
loadDataArrayList<String>() Response
loadQuokkaFacts() void
loadSettings() void
loadState() Response
removeCrypto(String) Response
removeFavorite(String, int) Response
removeStock(String) Response
saveAllCrypto() void
saveAllStock() void
saveData() Response
saveSettings() void
saveState() void
setActiveEntryList(ArrayList<Entry>) void
setActiveSymbol(String) void
setActiveType(int) void
setFileUrl(String) void
setNextActiveData() void
update(int, int) Response
updateAll() Response
updateAllCryptos() Response
updateAllStocks() Response
updateCrypto(int, int) Response
updateStock(int, int) Response
DataStorage
adapter StorageAdapter
DataStorage()
arrayListOfEntriesToArrayListOfStrings(ArrayList<Entry>) ArrayList<String>
init() void
load(ArrayList<String>) LoadData
loadSettings(ArrayList<String>) Map<String, String>
save(ArrayList<String>, ArrayList<Entry>) Response
saveSettings(Map<String, String>, ArrayList<String>) void
StorageAdapter
loadArrayList<String>() LoadData
loadSettings(ArrayList<String>) Map<String, String>
saveSettings(Map<String, String>, ArrayList<String>) void
storeArrayList<String>, ArrayList<String>) Response
LocalStorageTranslator
LocalStorageTranslator()
exists(Path) boolean
getFileNameArrayList<String>() String
getPath(String) Path?
loadArrayList<String>() LoadData
loadSettings(ArrayList<String>) Map<String, String>
saveSettings(Map<String, String>, ArrayList<String>) void
storeArrayList<String>, ArrayList<String>) Response
Stocks
adapter StocksAdapter
Stocks()
getStocks(String, int, int) ArrayList<Entry>
getCallIntervals() String[]
getCallTypes() String[]
init() void
StocksAdapter
getCallIntervals() String[]
getCallTypes() String[]
getStocksData(String, int, int) ArrayList<Entry>
AVAPIStocksTranslator
AVAPIStocksTranslator()
getCallIntervals() String[]
getCallTypes() String[]
getStockURL(String, CallInterval, CallType) URL
getStocksData(String, int, int) ArrayList<Entry>
parseStock(JSONJSONObject, CallType) ArrayList<Entry>
translateStock(CallType(CallType) String?
translateStock(JSONInterval(CallInterval) String?
translateStock(JSONInterval(CallType) String?
AlphaVantageConnection
baseURLAlpha String
apiKeyAlpha String
AlphaVantageConnection()
getJSONAURL() JSONObject?
translateIntervalEnum(int) CallInterval?
translateIntradayInterval(CallInterval) String?
translateTypeEnum(int) CallType?
Cryptos
adapter CryptosAdapter
Cryptos()
getCrypto(String, String, int) ArrayList<Entry>
getCallMarkets() String[]
getCallTypes() String[]
init() void
CryptosAdapter
getCallMarkets() String[]
getCallTypes() String[]
getCryptoData(String, String, int) ArrayList<Entry>
AVAPICryptoTranslator
AVAPICryptoTranslator()
getCallMarkets() String[]
getCallTypes() String[]
getCryptoData(String, String, int) ArrayList<Entry>
getCryptoURL(String, String, CallType) URL?
parseCrypto(JSONJSONObject, String) ArrayList<Entry>
translateCryptoCallType(CallType) String?
translateCryptoJSONInterval(CallType) String?
Qfacts
adapter QfactsAdapter
Qfacts()
getList(int) ArrayList<String>
init() void
QfactsAdapter
getQfactsList(int) ArrayList<String>
RandomFactsAPITranslator
baseURLQuokkas String
apiKeyQuokkas String
RandomFactsAPITranslator()
getQfactsList(int) ArrayList<String>
getQuokkaFact() String?
MaxProfit
MaxProfit()
createPriceChangeArray(double[]) double[]
findMaxCrossingProfit(double[], int, int, int) jump
findMaxProfit(double[], int, int) jump
getMaxProfit(ArrayList<Entry>) String[]
DataSegment
fileUrl String
symbol String
callType int
entryList ArrayList<Entry>
DataSegment()
getCallType() int
getEntryList() ArrayList<Entry>
getFileUrl() String
getSymbol() String
setCallType(int) void
setEntryList(ArrayList<Entry>) void
setFileUrl(String) void
setSymbol(String) void
CryptoDataSegment
callMarket int
CryptoDataSegment(String, int, int)
getCallMarket() int
setCallMarket(int) void
StockDataSegment
callInterval int
StockDataSegment(String, int, int)
getCallInterval() int
setCallInterval(int) void
Entry
formatter DateFormat
readableFormatter DateFormat
graphFormatter DateFormat
open double
close double
low double
high double
volume int
dateTime Date
Entry()
Entry(double, double, double, double, int, Date)
compareTo(Entry) int
getClose() double
getCloseString() String
getDateTime() Date
getDateTimeString() String
getFormatter() DateFormat
getGraphTimeString() String
getHigh() double
getHighString() String
getLow() double
getLowString() String
getOpen() double
getOpenString() String
getVolume() int
getVolumeString() String
setClose(double) void
setDateTime(Date) void
setHigh(double) void
setLow(double) void
setOpen(double) void
setVolume(int) void
toString() String
Powered by yFiles