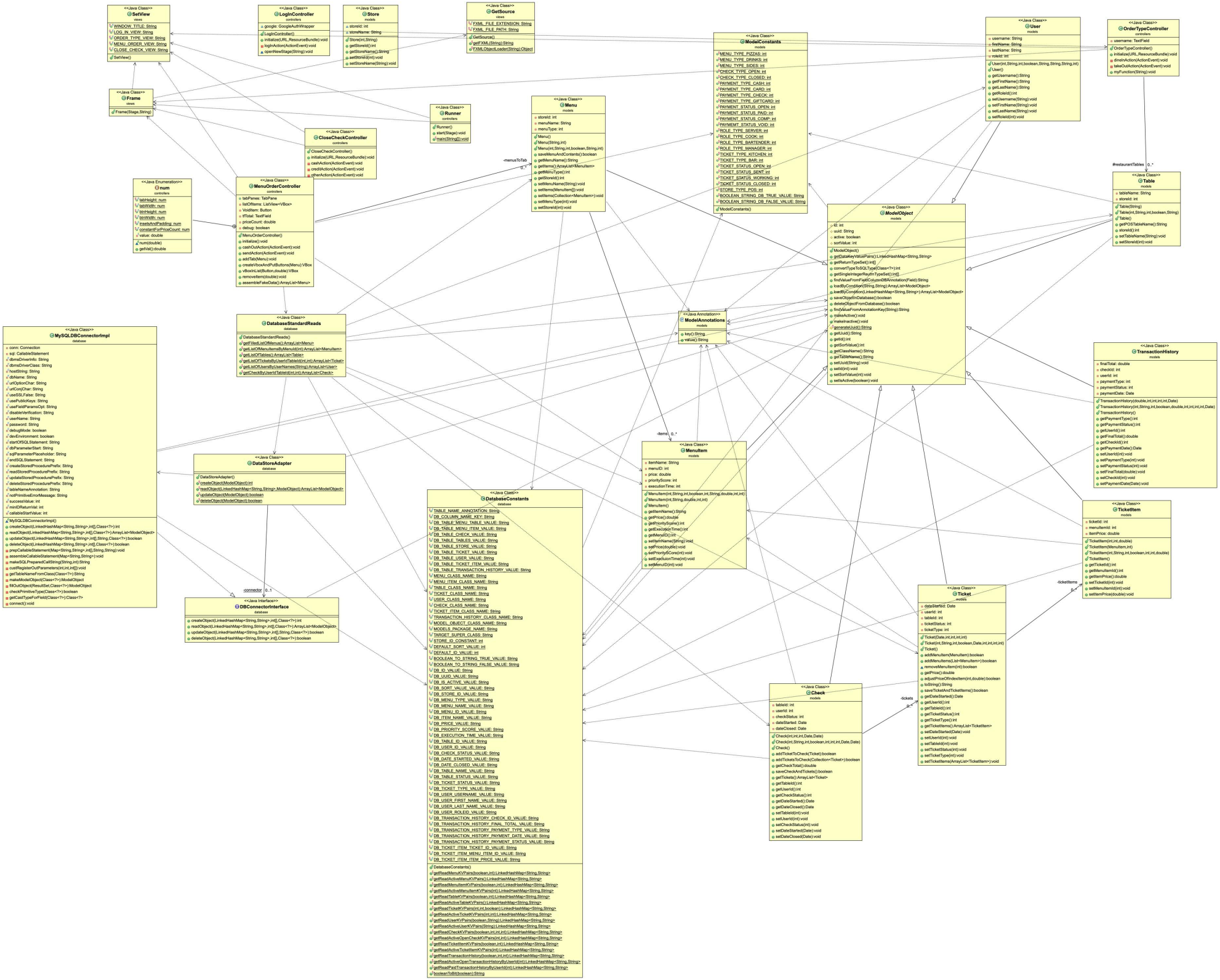
SetView
views
WINDOW_TITLE: String
LOG_IN_VIEW: String
ORDER_TYPE_VIEW: String
MENU_ORDER_VIEW: String
CLOSE_CHECK_VIEW: String
SetView()
LoginController
controllers
google: GoogleAuthWrapper
LoginController()
initialize(URL,ResourceBundle):void
logInAction(ActionEvent):void
openNewStage(String):void
Store
models
storeId: int
storeName: String
Store(int,String)
getStoreId():int
getStoreName():String
setStoreId(int):void
setStoreName(String):void
GetSource
views
FXML_FILE_EXTENSION: String
FXML_FILE_PATH: String
GetSource()
getFXML(String):String
FXMLObjectLoader(String):Object
ModelConstants
models
MENU_TYPE_PIZZAS: int
MENU_TYPE_DRINKS: int
MENU_TYPE_SIDES: int
CHECK_TYPE_OPEN: int
CHECK_TYPE_CLOSED: int
PAYMENT_TYPE_CASH: int
PAYMENT_TYPE_CARD: int
PAYMENT_TYPE_CHECK: int
PAYMENT_TYPE_GIFTCARD: int
PAYMENT_STATUS_OPEN: int
PAYMENT_STATUS_PAID: int
PAYMENT_STATUS_COMP: int
PAYMENT_STATUS_VOID: int
ROLE_TYPE_SERVER: int
ROLE_TYPE_COOK: int
ROLE_TYPE_BARTENDER: int
ROLE_TYPE_MANAGER: int
TICKET_TYPE_KITCHEN: int
TICKET_TYPE_BAR: int
TICKET_STATUS_OPEN: int
TICKET_STATUS_SENT: int
TICKET_STATUS_WORKING: int
TICKET_STATUS_CLOSED: int
STORE_TYPE_POS: int
BOOLEAN_STRING_DB_TRUE_VALUE: String
BOOLEAN_STRING_DB_FALSE_VALUE: String
ModelConstants()
User
models
username: String
firstName: String
lastName: String
roleId: int
User(int,String,int,boolean,String,String,String,int)
User()
getUsername():String
getFirstName():String
getLastName():String
getRoleId():int
setUsername(String):void
setFirstName(String):void
setLastName(String):void
setRoleId(int):void
OrderTypeController
controllers
username: TextField
OrderTypeController()
initialize(URL,ResourceBundle):void
dineInAction(ActionEvent):void
takeOutAction(ActionEvent):void
myFunction(String):void
Frame
views
Frame(Stage,String)
Runner
controllers
Runner()
start(Stage):void
main(String[]):void
CloseCheckController
controllers
CloseCheckController()
initialize(URL,ResourceBundle):void
cashAction(ActionEvent):void
creditAction(ActionEvent):void
otherAction(ActionEvent):void
Menu
models
storeId: int
menuName: String
menuType: int
Menu()
Menu(String,int)
Menu(int,String,int,boolean,String,int)
saveMenuAndContents():boolean
getMenuName():String
getItems():ArrayList<MenuItem>
getMenuType():int
getStoreId():int
setMenuName(String):void
setItems(MenuItem[]):void
setItems(Collection<MenuItem>):void
setMenuType(int):void
setStoreId(int):void
-menusToTab
0..*
#restaurantTables
0..*
Table
models
tableName: String
storeId: int
Table(String)
Table(int,String,int,boolean,String)
Table()
getPOSTableName():String
storeId():int
setTableName(String):void
setStoreId(int):void
num
controllers
tabHeight: num
tabWidth: num
btnHeight: num
btnWidth: num
insetsAndPadding: num
constantFerPriceCount: num
value: double
num(double)
getVal():double
MenuOrderController
controllers
tabPanes: TabPane
listOfItems: ListView<VBox>
VoidItem: Button
tfTotal: TextField
priceCount: double
debug: boolean
MenuOrderController()
initialize():void
cashOutAction(ActionEvent):void
sendAction(ActionEvent):void
addTab(Menu):void
createVboxAndPutButtons(Menu):VBox
vBoxInList(Button,double):VBox
removeItem(double):void
assembleFakeData():ArrayList<Menu>
ModelObject
models
id: int
uuid: String
active: boolean
sortValue: int
ModelObject()
getDataKeyValuePairs():LinkedHashMap<String,String>
getReturnTypeSet():int[]
convertTypeToSQLType(Class<?>):int
getSingleIntegerRetrnTypeSet():int[]
findValueFromFieldColumnDBAnnotation(Field):String
loadByCondition(String,String):ArrayList<ModelObject>
loadByCondition(LinkedHashMap<String,String>):ArrayList<ModelObject>
saveObjetInDatabase():boolean
deleteObjectFromDatabase():boolean
findValueFromAnnotationKey(String):String
makeActive():void
makeInactive():void
generateUuid():String
getUuid():String
getId():int
getSortValue():int
getClassName():String
getTableName():String
setUuid(String):void
setId(int):void
setSortValue(int):void
setIsActive(boolean):void
DatabaseStandardReads
database
DatabaseStandardReads()
getFiledListOfMenus():ArrayList<Menu>
getListOfMenuItemsByMenuId(int):ArrayList<MenuItem>
getListOfTables():ArrayList<Table>
getListOfTicketsByUserId/TableId(int,int):ArrayList<Ticket>
getListOfUsersByUserNames(String):ArrayList<User>
getCheckByUserS/TableId(int,int):ArrayList<Check>
ModelAnnotations
models
key():String
value():String
TransactionHistory
models
finalTotal: double
checkId: int
userId: int
paymentType: int
paymentStatus: int
paymentDate: Date
TransactionHistory(double,int,int,int,int,Date)
TransactionHistory(int,String,int,boolean,double,int,int,int,int,Date)
TransactionHistory()
getPaymentType():int
getPaymentStatus():int
getUserId():int
getFinalTotal():double
getCheckId():int
getPaymentDate():Date
setUserId(int):void
setPaymentType(int):void
setPaymentStatus(int):void
setFinaTotal(double):void
setCheckId(int):void
setPaymentDate(Date):void
MySQLDBConnectorImpl
database
conn: Connection
sql: CallableStatement
dbmsDriverInfo: String
dbmsDriverClass: String
hostString: String
dbName: String
urlOptionChar: String
urlConjChar: String
useSSLFalse: String
usePublicKeys: String
useFieldParamsOpt: String
disableVerification: String
userName: String
password: String
debugMode: boolean
devEnvironment: boolean
startDfSQLStatement: String
dbParameterStart: String
sqlParameterPlaceholder: String
endSQLStatement: String
createStoredProcedurePrefix: String
readStoredProcedurePrefix: String
updateStoredProcedurePrefix: String
deleteStoredProcedurePrefix: String
tableNameAnnotation: String
notPrimiteErrorMessage: String
successValue: int
minDfReturnVal: int
callableStartValue: int
MySQLDBConnectorImpl()
createObject(LinkedHashMap<String,String>,int[],Class<?>):int
readObject(LinkedHashMap<String,String>,int[],Class<?>):ArrayList<ModelObject>
updateObject(LinkedHashMap<String,String>,int[],String,Class<?>):boolean
deleteObject(LinkedHashMap<String,String>,int[],Class<?>):boolean
prepCallableStatement(Map<String,String>,int[],String,String):void
assembleCallableStatement(Map<String,String>):void
makeSQLPrepareCallString(String,int):String
custRegisterOutParameters(int,int,int[]):void
getTableNameFromClass(Class<?>):String
makeModelObject(Class<?>):ModelObject
fillOutObject(ResultSet,Class<?>):ModelObject
checkPrimitiveType(Class<?>):boolean
getCastTypeForField(Class<?>):Class<?>
connect():void
DataStoreAdapter
database
DataStoreAdapter()
createObject(ModelObject):int
readObject(LinkedHashMap<String,String>,ModelObject):ArrayList<ModelObject>
updateObject(ModelObject):boolean
deleteObject(ModelObject):boolean
-items
0..*
MenuItem
models
itemName: String
menuID: int
price: double
priorityScore: int
executionTime: int
MenuItem(int,String,int,boolean,int,String,double,int,int)
MenuItem(int,String,double,int,int)
MenuItem()
getItemName():String
getPrice():double
getPriorityScore():int
getExecutionTime():int
getMenuID():int
setItemName(String):void
setPrice(double):void
setPriorityScore(int):void
setExecutionTime(int):void
setMenuID(int):void
TicketItem
models
ticketId: int
menuItemId: int
itemPrice: double
TicketItem(int,int,double)
TicketItem(MenuItem,int)
TicketItem(int,String,int,boolean,int,int,double)
TicketItem()
getTicketId():int
getMenuItemId():int
getItemPrice():double
setTicketId(int):void
setMenuItemId(int):void
setItemPrice(double):void
-connector
0..1
DBConnectorInterface
database
DatabaseConstants
database
TABLE_NAME_ANNOTATION: String
DB_COLUMN_NAME_KEY: String
DB_TABLE' MENU_TABLE_VALUE: String
DB_TABLE' MENU_ITEM_VALUE: String
DB_TABLE_CHECK_VALUE: String
DB_TABLE_TABLES_VALUE: String
DB_TABLE_STORE_VALUE: String
DB_TABLE_TICKET_VALUE: String
DB_TABLE_USER_VALUE: String
DB_TABLE_TICKET_ITEM_VALUE: String
DB_TABLE_TRANSACTION_HISTORY_VALUE: String
MENU_CLASS_NAME: String
MENU_ITEM_CLASS_NAME: String
TABLE_CLASS_NAME: String
TICKET_CLASS_NAME: String
USER_CLASS_NAME: String
CHECK_CLASS_NAME: String
TICKET_ITEM_CLASS_NAME: String
TRANSACTION_HISTORY_CLASS_NAME: String
MODEL_OBJECT_CLASS_NAME: String
MODELS_PACKAGE_NAME: String
TARGET_SUPER_CLASS: String
STORE_ID_CONSTANT: int
DEFAULT_SORT_VALUE: int
DEFAULT_ID_VALUE: int
BOOLEAN_TO_STRING_TRUE_VALUE: String
BOOLEAN_TO_STRING_FALSE_VALUE: String
DB_ID_VALUE: String
DB_UUID_VALUE: String
DB_IS_ACTIVE_VALUE: String
DB_SORT_VALUE_VALUE: String
DB_STORE_ID_VALUE: String
DB_MENU_TYPE_VALUE: String
DB_MENU_NAME_VALUE: String
DB_MENU_ID_VALUE: String
DB_ITEM_NAME_VALUE: String
DB_PRICE_VALUE: String
DB_PRIORITY_SCORE_VALUE: String
DB_EXECUTION_TIME_VALUE: String
DB_TABLE_ID_VALUE: String
DB_USER_ID_VALUE: String
DB_CHECK_STATUS_VALUE: String
DB_DATE_STARTED_VALUE: String
DB_DATE_CLOSED_VALUE: String
DB_TABLE_NAME_VALUE: String
DB_TABLE_STATUS_VALUE: String
DB_TICKET_STATUS_VALUE: String
DB_TICKET_TYPE_VALUE: String
DB_USER_USERNAME_VALUE: String
DB_USER_FIRST_NAME_VALUE: String
DB_USER_LAST_NAME_VALUE: String
DB_USER_ROLE/ID_VALUE: String
DB_TRANSACTION_HISTORY_CHECK_ID_VALUE: String
DB_TRANSACTION_HISTORY_FINAL_TOTAL_VALUE: String
DB_TRANSACTION_HISTORY_PAYMENT_TYPE_VALUE: String
DB_TRANSACTION_HISTORY_PAYMENT_DATE_VALUE: String
DB_TRANSACTION_HISTORY_PAYMENT_STATUS_VALUE: String
DB_TICKET_ITEM_TICKET_ID_VALUE: String
DB_TICKET_ITEM_MENU_ITEM_ID_VALUE: String
DB_TICKET_ITEM_ITEM_PRICE_VALUE: String
DatabaseConstants()
getReadMenuKVPairs(boolean,int):LinkedHashMap<String,String>
getReadActiveMenuKVPairs():LinkedHashMap<String,String>
getReadMenuItemKVPairs(boolean,int):LinkedHashMap<String,String>
getReadActiveMenuItemKVPairs(int):LinkedHashMap<String,String>
getReadTableKVPairs(boolean,int):LinkedHashMap<String,String>
getReadActiveTableKVPairs():LinkedHashMap<String,String>
getReadTicketKVPairs(int,int,boolean):LinkedHashMap<String,String>
getReadActiveTicketKVPairs(int,int):LinkedHashMap<String,String>
getReadUserKVPairs(boolean,String):LinkedHashMap<String,String>
getReadActiveUserKVPairs(String):LinkedHashMap<String,String>
getReadCheckKVPairs(boolean,int,int,int):LinkedHashMap<String,String>
getReadActiveOpenCheckKVPairs(int,int):LinkedHashMap<String,String>
getReadTicketItemKVPairs(boolean,int):LinkedHashMap<String,String>
getReadActiveTicketItemKVPairs(int):LinkedHashMap<String,String>
getReadTransactionHistory(boolean,int,int):LinkedHashMap<String,String>
getReadActiveOpenTransactionHistoryByUserId(int):LinkedHashMap<String,String>
getReadPaidTransactionHistoryByUserId(int):LinkedHashMap<String,String>
booleanToBit(boolean):String
Ticket
models
dateStartId: Date
userId: int
tableId: int
ticketStatus: int
ticketType: int
Ticket(Date,int,int,int,int)
Ticket(int,String,int,boolean,Date,int,int,int,int)
Ticket()
addMenuItem(MenuItem):boolean
addMenuItems(List<MenuItem>):boolean
removeMenuItem(int):boolean
getPrice():double
adjustPriceOfIndexItem(int,double):boolean
toString():String
saveTicketAndTicketItems():boolean
getDateStarted():Date
getUserId():int
getTableId():int
getTicketStatus():int
getTicketType():int
getTicketItems():ArrayList<TicketItem>
setDateStarted(Date):void
setUserId(int):void
setTableId(int):void
setTicketStatus(int):void
setTicketType(int):void
setTicketItems(ArrayList<TicketItem>):void
-tickets
0..*
-ticketItems
0..*
Check
models
tableId: int
userId: int
checkStatus: int
dateStarted: Date
dateClosed: Date
Check(int,int,int,Date,Date)
Check(int,String,int,boolean,int,int,int,Date,Date)
Check()
addTicketToCheck(Ticket):boolean
addTicketsToCheck(Collection<Ticket>):boolean
getCheckTotal():double
saveCheckAndTickets():boolean
getTickets():ArrayList<Ticket>
getTableId():int
getUserId():int
getCheckStatus():int
getDateStarted():Date
getDateClosed():Date
setTableId(int):void
setUserId(int):void
setCheckStatus(int):void
setDateStarted(Date):void
setDateClosed(Date):void