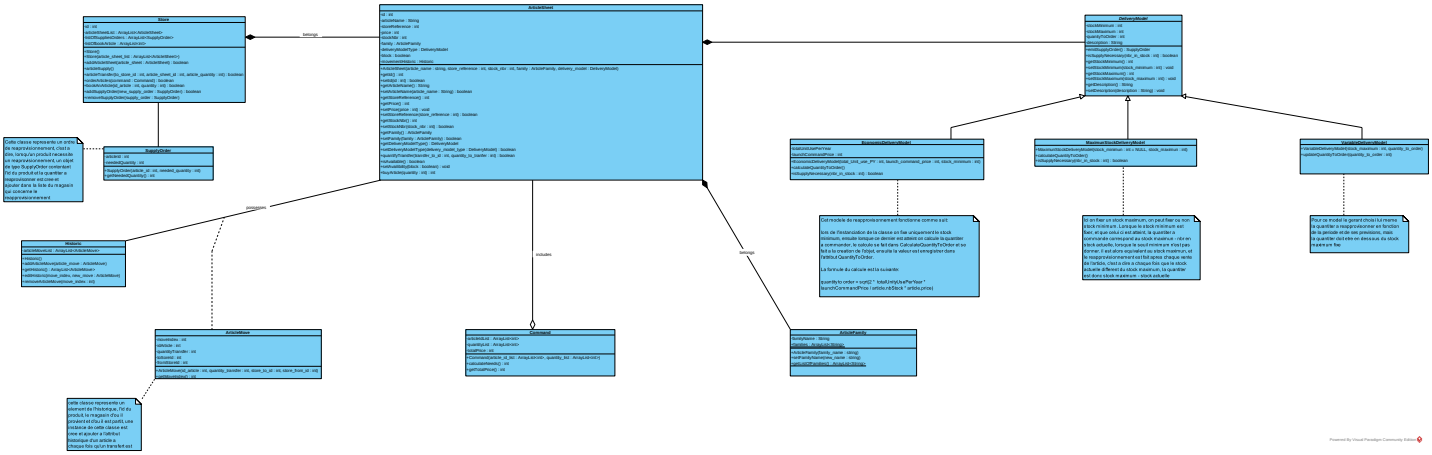
belongs
Store
id : int
articleSheetsList : ArrayList<ArticleSheet>
listOfSupplierOrders : ArrayList<SupplyOrder>
listOfBookArticle : ArrayList<>
+Store()
+getBookArticle_sheet_list() : ArrayList<ArticleSheet>()
+addArticleSheet(article_sheet : ArticleSheet) : boolean
+addInSupply()
+articleTransfer(to_store_id : int, article_sheet_id : int, article_quantity : int) : boolean
+isOrderAlreadyexecuted(Command) : boolean
+hasArticle(id_article : int, quantity : int) : boolean
+isAlreadyOrderExist(new_supply_order : SupplyOrder) : boolean
+listOfNeedSupplyOrderOfSupply_order() : ArrayList<>
belongs
ArticleSheet
id : int
articleName : String
storeReference : int
price : int
stockMin : int
stockMax : int
quantityAvailability
delivery_modelType : DeliveryModel
block : boolean
occurrenceWeeks : Weeks[]
+ArticleSheet(article_name : string, store_reference : int, stock_min : int, family : ArticleFamily, delivery_model : DeliveryModel)
+getId() : int
+setId(id : int) : boolean
+getArticleName() : String
+setArticleName(article_name : String) : boolean
+getStoreReference() : int
+getPrice() : int
+setPrice(price : int) : void
+setStoreReference(store_reference : int) : boolean
+getStockMin() : int
+setStockMin(stock_min : int) : boolean
+getFamily() : ArticleFamily
+setFamily(family : ArticleFamily) : boolean
+getDeliveryModelType() : DeliveryModel
+setDeliveryModelType(delivery_model_type : DeliveryModel) : boolean
+isAvailable(quantity_to_ask : int, quantity_to_transfer : int) : boolean
+isAvailable() : boolean
+setAvailable(block : boolean) : void
+buyArticle(quantity : int) : int
includes
Command
articleIdList : ArrayList<int>
quantityList : ArrayList<int>
+Command()
+Command(article_id_list : ArrayList<int>, quantity_list : ArrayList<int>)
+calculateMonthly() : int
+getTotalPrice() : int
assigns
ArticleFamily
familyName : String
[Section : ArrayList<String>]
+ArticleFamily(family_name : string)
+setFamilyName(new_name : String)
+getFamilyNameNew() : String
DeliveryModel
stockMinimum : int
stockMaximum : int
quantityToOrder : int
+DeliveryModel()
+checkSupplyToBuy() : SupplyOrder
+isSupplyNeeded(qty_to_stock : int) : boolean
+getStockMinimum() : int
+setStockMinimum(stock_minimum : int) : void
+getStockMaximum() : int
+setStockMaximum(stock_maximum : int) : void
+getDescription() : String
+setDescription(description : String) : void
BusinessDeliveryModel
+businessDelivery() : int
+businessDelivery(stock_min : int, launch_command_price : int, stock_minimum : int) : int
+calculateQuantityToOrder()
+isSupplyNeeded(qty_to_stock : int) : boolean
Cet modèle de réapprovisionnement fonctionne comme suit :
lors de l'initialisation de la classe on fixe uniquement le stock minimum. Lorsque le stock minimum est atteint, il faut commander. le calcule se fait dans CalculateQuantityToOrder et on fixe la la creation de l'objet, ensuite la valeur est enregistrer dans l'attribut QuantityToOrder.
La formule du calcule est la suivante:
quantity to order = sqrt(2 * totalUnitsyearPerYear * launchCommandPrice / article.stoBlock * article.price)
MaximumStockDeliveryModel
+maximumStockDelivery(stock_minimum : int = NULL, stock_maximum : int)
+calculateQuantityToOrder()
+isSupplyNeeded(qty_to_stock : int) : boolean
Ici on fixe un stock maximum, on peut fixer ou non stock minimum. Lorsque le stock minimum est fixé, et que celui-ci est atteint, la quantité à commander est calculée de sorte à ne pas avoir un stock actuelle, lorsque le seuil minimum n'est pas donner, il est alors equivalent au stock maximum, et le reapprovisionnement est fait apres chaque vente de l'article. C'est à dire a chaque fois que le stock actuelle different du stock maximum, la quantité est donc stock maximum - stock actuelle
VariableDeliveryModel
+variableDelivery(stock_minimum : int, quantity_to_order : int)
+calculateQuantityToOrder(quantity_to_order : int)
Pour ce model le gerant choisi lui meme la quantité a reapprovisionner en fonction de la periode et de ses possesions, mais la quantité n'est elle en aucun cas du stock inferieur au flux
Weeks
articleWeeksList : ArrayList<ArticleWeek>
+Weeks()
+addArticleWeek(article_move : ArticleWeek)
+getWeeks() : ArrayList<ArticleWeek>
+addWeeks(move_index, new_move : ArticleWeek)
+removeArticleWeek(move_index : int)
ArticleWeek
moveIndex : int
articleId : int
quantityMoved : int
[source : int]
[destination : int]
+ArticleWeek()
+ArticleWeek(id_article : int, quantity_transfer : int, store_to_id : int, store_from_id : int)
+getWeekIndex() : int
Cette classe represente un element de l'historique, l'id du produit, le magasin d'ou il est transferé et la quantité. Une instance de cette classe est une etiquette a l'attribut historique d'un article a chaque fois qu'un transfert est
SupplyOrder
articleId : int
neededQuantity : int
+SupplyOrder(article_id : int, needed_quantity : int)
+getNeededQuantity() : int
Cette classe represente un article de magasin environnement, c'est à dire, lorsque le stock de l'article est inférieur au seuil minimum, un objet de type SupplyOrder correspondant au produit et la quantité a reapprovisionner est cree et ajouter dans la liste du magasin qui contenant le reapprovisionnement
Powered By Visual Paradigm Community Edition