SDE

- # instance: SDE
- # db: mysqli
- # numQueries: int
- # timeQueries: float
- + instance(db: mysqli) : SDE
- # __construct(db: mysqli) : SDE
- + query(sql: string) : mysql_result
- + multiQuery(sql: string) : bool
- + flushDbResults(): void
- + commit(): bool + rollback(): bool
- # addQueryTime(): void
- + getStats() : array
- + makeUpsertQuery(table: string, insert: array, update: array) :...
- + makeUdateQuery(table: string, update: array, where: array) :...
- + sanitizeString(string: string) : string
- + sanitizeAndEnquoteString(string: string) : string

Util

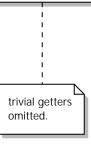
- + quantitiesToReadable(val: int) : string
- + secondsToReadable(fseconds: int) : string

FitParser

- + parseEftFit(eftFit: string) : MaterialParseResult
- + parseXmlFit(fitDom: DOMDocument) : MaterialParseResult
- + parseScanResult(scanResult: string): MaterialParseResult

Config

- # sdeDbHost: string
- # sdeDbPort: int
- # sdeDbUser: string
- # sdeDbPw: string
- # sdeDbName: string
- # iveeDbName: string
- # cachePrefix: string
- # cacheHost: string
- # cachePort: int
- # emdrRelayUrl: string
- # crestBaseUrl: string
- # userAgent: string
- # classes: array
- + getIveeClassName(classNickname: string) : string



Exception classes omitted.

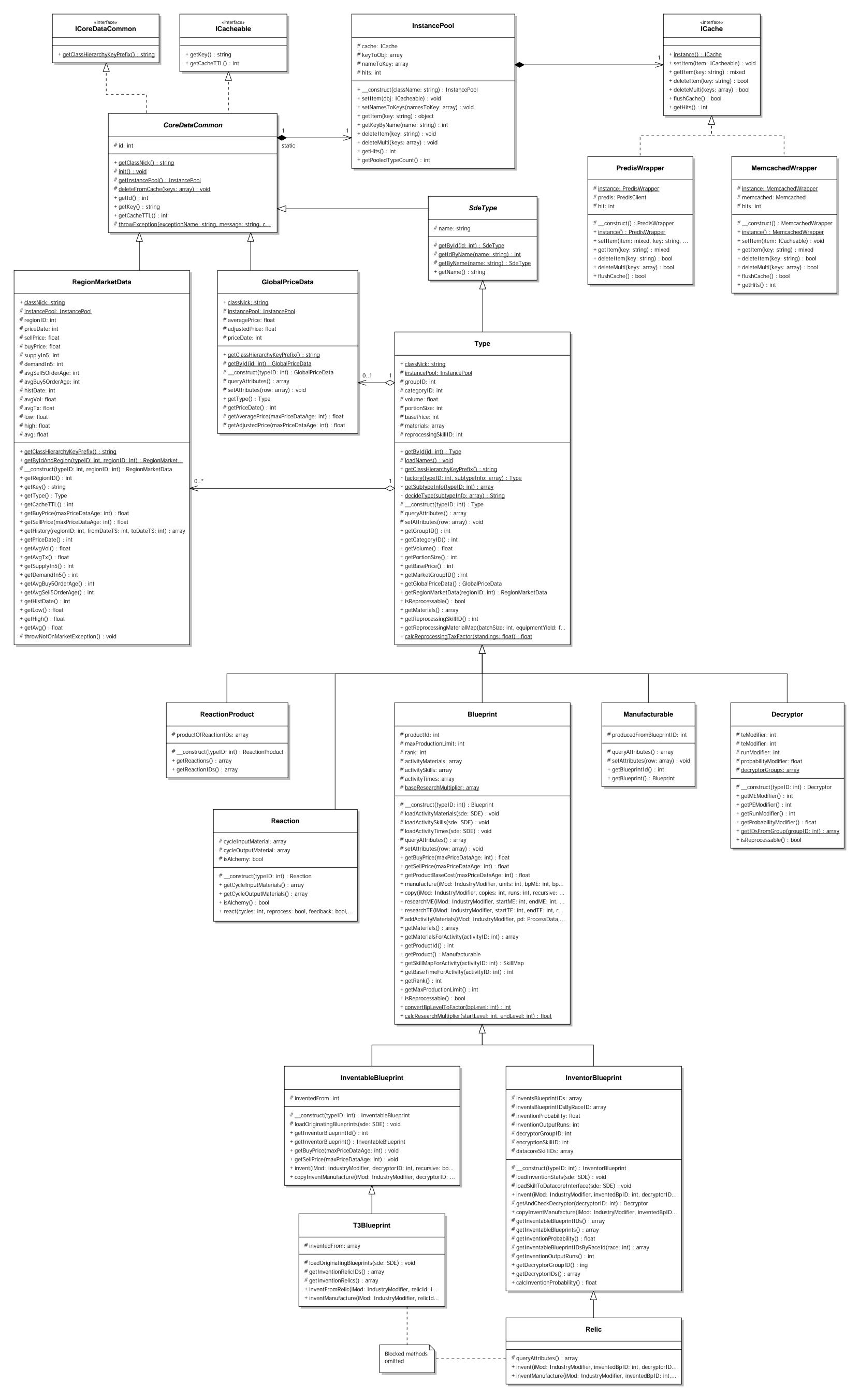
Defaults

- # instance: Defaults
- # defaultRegionID: int
- # trackedMarketRegionIDs: array
- # defaultBuyTaxFactor: float
- # defaultSellTaxFactor: float
- # defaultBpoMe: int
- # defaultBpoMe: int
- # maxPriceDataAge: int
- # bpMeLevels: array
- # bpTeLevels: array
- + instance(): Defaults
- + getSkillLevel(skillID: int) : int
- + getIndustryImplantTimeModifiers() : array
- + setBpMeLevel() : void
- + setBpTeLevel(): void

MyDefaults

trivial getters ommited.

For user implementation. Also serves as example for the intended way of extending iveeCore



ProcessData

- # activity: int
- # producesTypeID: int
- # producesQuantity: int
- # processTime: int
- # processCost: float
- # assemblyLineID: int
- # solarSystemID: int
- # skills: SkillMap
- # materials: MaterialMap
- # subProcessData: array
- + __construct(producesTypeID: int, producesQuantity: int...
- + addMaterial(typeID: int, amount: int) : void
- + addSkill(skillID: int, level: int) : void
- + addSkillMap(sm: SkillMap) : void
- + addSubProcessData(subProcessData: ProcessData) : void
- + getActivityID(): int
- + getProducedType() : Type
- + getNumProducedUnits(): int
- + getSubProcesses() : array
- + getProcessCost() : float
- + getSolarSystemID(): int
- + getAssemblyLineTypeID(): int
- + getTotalProcessCost() : float
- + getMaterialBuyCost(maxPriceDataAge: int, regionId: int...
- + getTotalMaterialBuyCost(maxPriceDataAge: int, regionI...
- + getTotalCost(maxPriceDataAge: int, regionId: int) : float
- + getMaterialMap() : MaterialMap
- + getTotalMaterialMap(): MaterialMap
- + getMaterialVolume(): float
- + getTotalMaterialVolume() : float
- + getSkillMap() : SkillMap
- + getTotalSkillMap() : SkillMap
- + getTime() : int
- + getTotalTime() : int
- + getTotalTimes() : array
- + getTotalProfit(maxPriceDataAge: int, regionId: int) : float
- + printData() : void

\ _{0..}*

ReactionProcessData

+ __construct(inputMaterialMap: MaterialMap, ou...

+ getInputBuyCost(maxPriceDataAge: int) : float

+ getOutputBuyValue(maxPriceDataAge: int) : float

inputMaterialMap: MaterialMap

cycles: float

withRefining: bool # withFeedback: bool

+ getCycles() : float + getTime() : float

+ withRefining(): bool

+ withFeedback(): bool

outputMaterialMap: MaterialMap

+ geInputMaterialMap() : MaterialMap

+ getOutputMaterialMap(): MaterialMap

+ getProfit(maxPriceDataAge: int) : float

- # startMELevel: int
- # endMELevel: int
- + getStartMELevel() : int
- + getEndMELevel(): int

CopyProcessData

ManufactureProcessData

+ __construct(producesTypeID: int, producesQua...

+ getTotalCostPerUnit(maxPriceDataAge: int, regi...

 $+ getTotalProfit (maxPriceDataAge: int, \ regionId: i...$

outputRuns: int

bpMeLevel: int

bpPeLevel: int

+ getMeLevel() : int

+ getPeLevel() : int

+ printData() : void

+ getSlotCost() : float

- + __construct(bpCopyID: int, copyQuantity: int, o...
- + getOutputRuns(): int

InventionProcessData

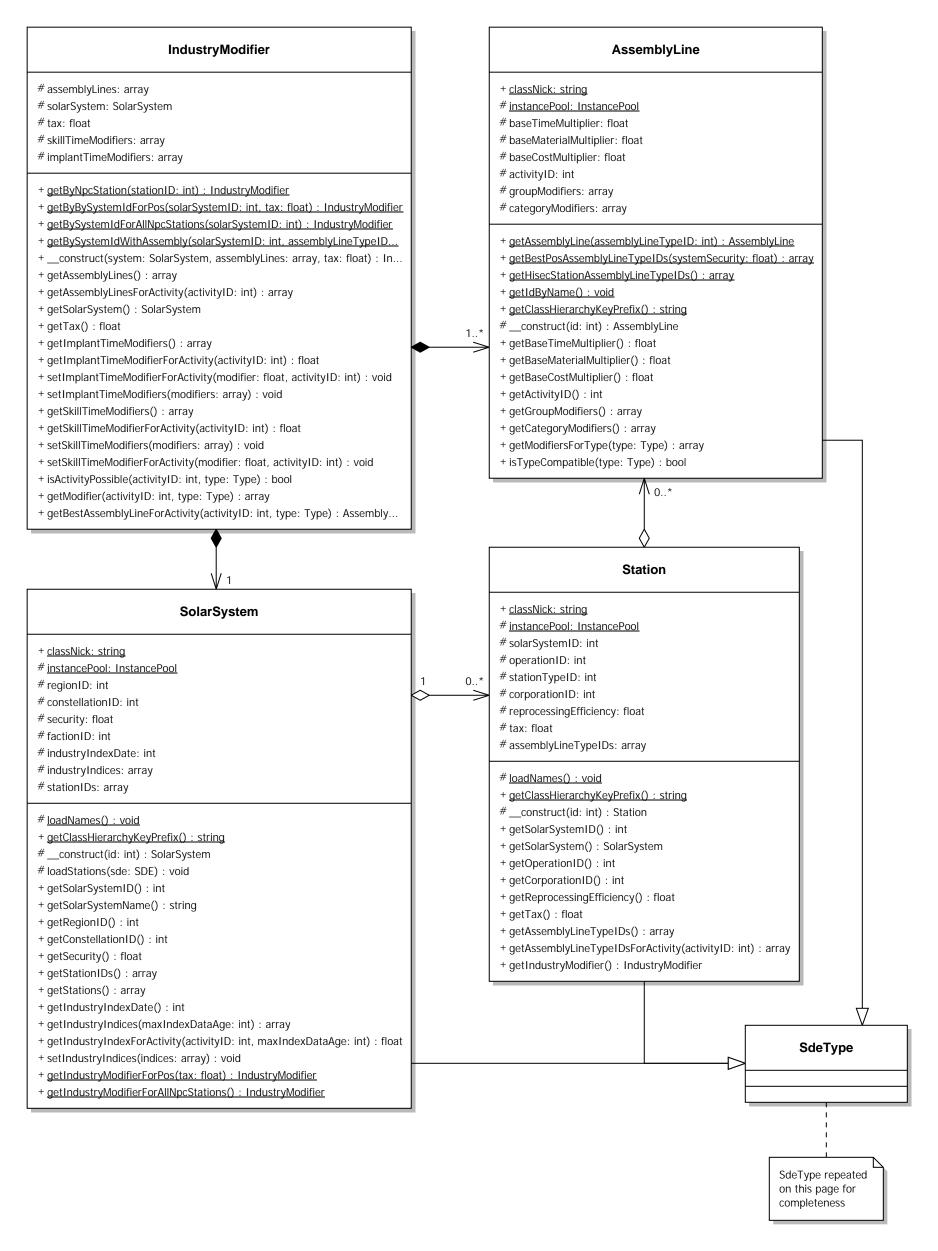
- # probability: float
- # resultRuns: int
- # resultME: int
- # resultTE: int
- + __construct(producesTypeID: int, inventionTime: int, proce...
- + getResultRuns(): int
- + getResultME(): int
- + getResultPE(): int
- + getProbability() : float
- + getSuccessTime() : float
- + getTotalSuccessTime() : float
- + getTotalSuccessTimes() : array
- + getSuccessMaterialMap(): MaterialMap
- + getTotalSuccessMaterialMap() : MaterialMap
- + getSuccessMaterialVolume() : float
- + getTotalSuccessMaterialVolume(): float
- + getProcessCost() : float
- + getSuccessProcessCost() : float
- + getTotalSuccessProcessCost() : float
- + getSuccessMaterialBuyCost(maxPriceDataAge: int, regionId:...
- + getTotalSuccessMaterialBuyCost(maxPriceDataAge: int, regi...
- + getTotalSuccessCost(maxPriceDataAge: int, regionId: int): ...
- + printData(): void

ResearchMEProcessData

- + __construct(researchedBpID: int, researchTime: i...

ResearchTEProcessData

- # startTELevel: int
- # endTELevel: int
- + __construct(researchedBpID: int, researchTime: i...
- + getStartTELevel(): int
- + getEndTELevel(): int



MaterialMap

materials: array

+ addMaterial(typeID: int, quatity: int) : void

+ addMaterials(materials: array): void

+ subtractMaterial(typeID: int, quantity: int): void

+ symmetricDifference(m1: MaterialMap, m2: MaterialMap) : void

+ addMaterialMap(materials: MaterialMap) : void

+ getMaterials() : array

+ getMultipliedMaterialMap(factor: float) : MaterialMap

+ reprocessMaterials(equipmentYield: float, reprocessingTaxFa...

+ getMaterialVolume() : float

+ getMaterialBuyCost(maxPriceDataAge: int, regionId: int): float

+ getMaterialSellValue(maxPriceDataAge: int, regionId: int) : fl...

MaterialParseResult

unparseables: array

+ addUnparseable(unparseable: string) : void

+ getUnparseables() : array

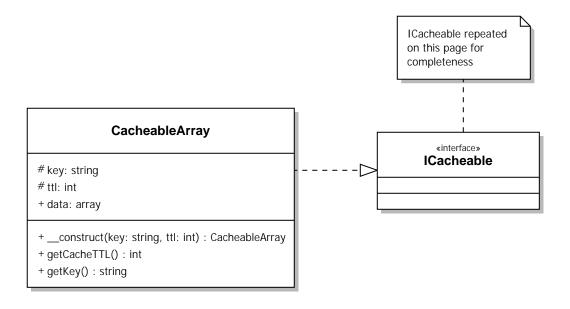
SkillMap

SkillMap: array

+ sanityCheckSkillLevel(skillLevel: int) : bool

+ addSkill(skillID: int, level: int) : void + addSkillMap(skillMap: SkillMap) : void

+ getSkills() : array



EmdrConsumer

- # instance: EmdrConsumer
- # trackedTypeIDs: array
- # trackedMarketRegionIDs: array
- # regions: array # sde: SDE
- # cache: ICache
- # emdrPriceUpdateClass: string
 # emdrHistoryUpdateClass: string
- + instance() : EmdrConsumer
- # __construct() : EmdrConsumer
- + run() : void
- # handleMarketData(marketData: stdClass) : void
- # filterData(typeID: int, regionID: int, generatedAt: int): bool
- # getTimestamps() : array
- # getTimestampsDB() : array
- # updateCaches(): void
- # handleOrderData(): void
- # handleHistoryData(): void
- + getTypeNameByID() : string
- + getRegionNameByID() : string

CrestFetcher

- $\# \ baseUrl: \ string$
- # userAgent: string
- + __construct() : Fetcher
- + getCrestData(path: string, representationName: string): stdClass
- $\#\, curl Get Json (url:\, string)\,:\, std Class$
- # parseContentTypeToRepresentation(contentType: string) : string

EmdrPriceUpdate

- # typeID: int
- # regionID: int
- # generatedAt: int
 # averages: array
- # sell: float
- # avgSell5OrderAge: int
- # buy: float
- # avgBuy5OrderAge: int
- $\#\,demandIn5\colon int$
- # supplyIn5: int
- + __construct(typeID: int, regionID: int, gen...
- # cmp(a: array, b: array) : int
- + insertIntoDB(): void
- # getPriceStats(odata: array, averages: arra...

EmdrHistoryUpdate

- # typeID: int
- # regionID: int
- # generatedAt: int
- # rows: array
- + __construct(typeID: int, regionID: int, gen...
- + insertIntoDB(): void

CrestDataUpdater

- # path: string
- # representationName: string
- # data: stdClass
- $\#\, updatedIDs \colon array$
- $+ \underline{\hspace{1.5cm}} construct (data: stdClass): CrestDataUpdater \\$
- + insertIntoDB(): void
- + process DataItem To SQL (item: stdClass): string
- # invalidateCaches(): void
- + doUpdate(): void

IndustryFacilitiesUpdater IndustrySystemsUpdater MarketPricesUpdater