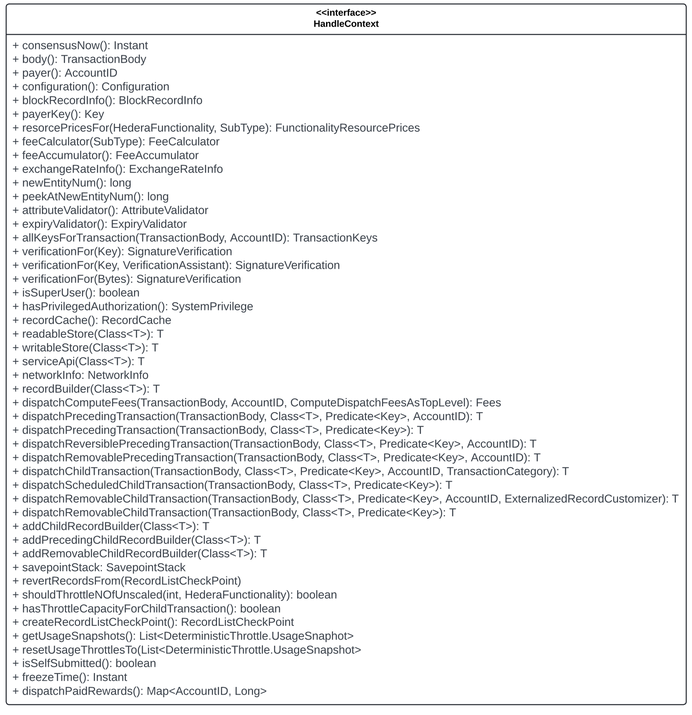
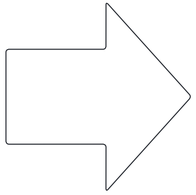
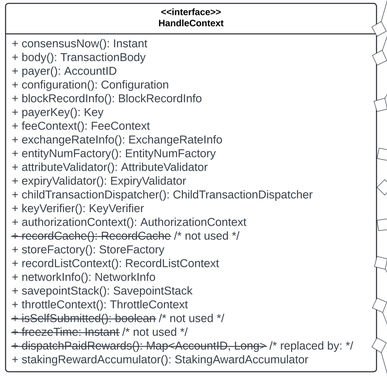
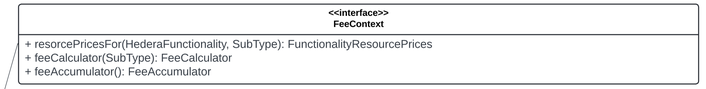
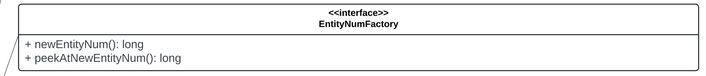
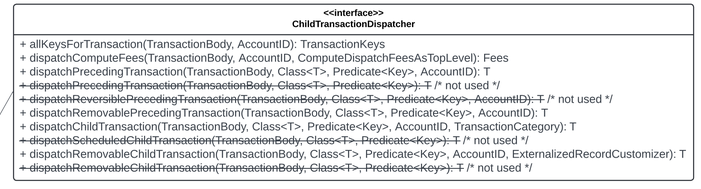

<<interface>>
HandleContext
+ consensusNow(): Instant
+ body(): TransactionBody
+ payer(): AccountID
+ configuration(): Configuration
+ blockRecordInfo(): BlockRecordInfo
+ payerKey(): Key
+ resorcePricesFor(HederaFunctionality, SubType): FunctionalityResourcePrices
+ feeCalculator(SubType): FeeCalculator
+ feeAccumulator(): FeeAccumulator
+ exchangeRateInfo(): ExchangeRateInfo
+ newEntityNum(): long
+ peekAtNewEntityNum(): long
+ attributeValidator(): AttributeValidator
+ expiryValidator(): ExpiryValidator
+ allKeysForTransaction(TransactionBody, AccountID): TransactionKeys
+ verificationFor(Key): SignatureVerification
+ verificationFor(Key, VerificationAssistant): SignatureVerification
+ verificationFor(Bytes): SignatureVerification
+ isSuperUser(): boolean
+ hasPrivilegedAuthorization(): SystemPrivilege
+ recordCache(): RecordCache
+ readableStore(Class<T>): T
+ writableStore(Class<T>): T
+ serviceApi(Class<T>): T
+ networkInfo: NetworkInfo
+ recordBuilder(Class<T>): T
+ dispatchComputeFees(TransactionBody, AccountID, ComputeDispatchFeesAsTopLevel): Fees
+ dispatchPrecedingTransaction(TransactionBody, Class<T>, Predicate<Key>, AccountID): T
+ dispatchPrecedingTransaction(TransactionBody, Class<T>, Predicate<Key>): T
+ dispatchReversiblePrecedingTransaction(TransactionBody, Class<T>, Predicate<Key>, AccountID): T
+ dispatchRemovablePrecedingTransaction(TransactionBody, Class<T>, Predicate<Key>, AccountID): T
+ dispatchChildTransaction(TransactionBody, Class<T>, Predicate<Key>, AccountID, TransactionCategory): T
+ dispatchScheduledChildTransaction(TransactionBody, Class<T>, Predicate<Key>): T
+ dispatchRemovableChildTransaction(TransactionBody, Class<T>, Predicate<Key>, AccountID, ExternalizedRecordCustomizer): T
+ dispatchRemovableChildTransaction(TransactionBody, Class<T>, Predicate<Key>): T
+ addChildRecordBuilder(Class<T>): T
+ addPrecedingChildRecordBuilder(Class<T>): T
+ addRemovableChildRecordBuilder(Class<T>): T
+ savepointStack: SavepointStack
+ revertRecordsFrom(RecordListCheckPoint)
+ shouldThrottleNOfUnscaled(int, HederaFunctionality): boolean
+ hasThrottleCapacityForChildTransaction(): boolean
+ createRecordListCheckPoint(): RecordListCheckPoint
+ getUsageSnapshots(): List<DeterministicThrottle.UsageSnaphot>
+ resetUsageThrottlesTo(List<DeterministicThrottle.UsageSnapshot>
+ isSelfSubmitted(): boolean
+ freezeTime(): Instant
+ dispatchPaidRewards(): Map<AccountID, Long>

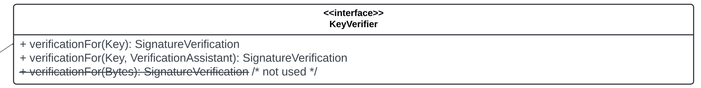



<<interface>>
HandleContext
+ consensusNow(): Instant
+ body(): TransactionBody
+ payer(): AccountID
+ configuration(): Configuration
+ blockRecordInfo(): BlockRecordInfo
+ payerKey(): Key
+ feeContext(): FeeContext
+ exchangeRateInfo(): ExchangeRateInfo
+ entityNumFactory(): EntityNumFactory
+ attributeValidator(): AttributeValidator
+ expiryValidator(): ExpiryValidator
+ childTransactionDispatcher(): ChildTransactionDispatcher
+ keyVerifier(): KeyVerifier
+ authorizationContext(): AuthorizationContext
+ recordCache(): RecordCache /* not used */
+ storeFactory(): StoreFactory
+ recordListContext(): RecordListContext
+ networkInfo(): NetworkInfo
+ savepointStack(): SavepointStack
+ throttleContext(): ThrottleContext
+ isSelfSubmitted(): boolean /* not used */
+ freezeTime: Instant /* not used */
+ dispatchPaidRewards(): Map<AccountID, Long> /* replaced by: */
+ stakingRewardAccumulator(): StakingAwardAccumulator


<<interface>>
FeeContext
+ resorcePricesFor(HederaFunctionality, SubType): FunctionalityResourcePrices
+ feeCalculator(SubType): FeeCalculator
+ feeAccumulator(): FeeAccumulator

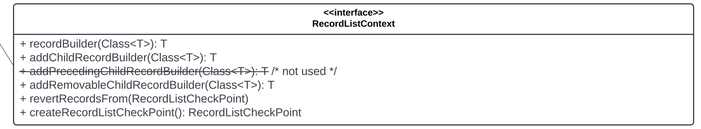

<<interface>>
EntityNumFactory
+ newEntityNum(): long
+ peekAtNewEntityNum(): long

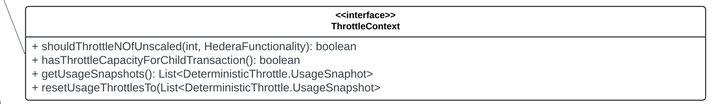

<<interface>>
ChildTransactionDispatcher
+ allKeysForTransaction(TransactionBody, AccountID): TransactionKeys
+ dispatchComputeFees(TransactionBody, AccountID, ComputeDispatchFeesAsTopLevel): Fees
+ dispatchPrecedingTransaction(TransactionBody, Class<T>, Predicate<Key>, AccountID): T
+ dispatchPrecedingTransaction(TransactionBody, Class<T>, Predicate<Key>): T /* not used */
+ dispatchReversiblePrecedingTransaction(TransactionBody, Class<T>, Predicate<Key>, AccountID): T /* not used */
+ dispatchRemovablePrecedingTransaction(TransactionBody, Class<T>, Predicate<Key>, AccountID): T
+ dispatchChildTransaction(TransactionBody, Class<T>, Predicate<Key>, AccountID, TransactionCategory): T
+ dispatchScheduledChildTransaction(TransactionBody, Class<T>, Predicate<Key>): T /* not used */
+ dispatchRemovableChildTransaction(TransactionBody, Class<T>, Predicate<Key>, AccountID, ExternalizedRecordCustomizer): T
+ dispatchRemovableChildTransaction(TransactionBody, Class<T>, Predicate<Key>): T /* not used */


<<interface>>
KeyVerifier
+ verificationFor(Key): SignatureVerification
+ verificationFor(Key, VerificationAssistant): SignatureVerification
+ verificationFor(Bytes): SignatureVerification /* not used */


<<interface>>
AuthorizationContext
+ isSuperUser(): boolean
+ hasPrivilegedAuthorization(): SystemPrivilege


<<interface>>
StoreFactory
+ readableStore(Class<T>): T
+ writableStore(Class<T>): T
+ serviceApi(Class<T>): T


<<interface>>
RecordListContext
+ recordBuilder(Class<T>): T
+ addChildRecordBuilder(Class<T>): T
+ addPrecedingChildRecordBuilder(Class<T>): T /* not used */
+ addRemovableChildRecordBuilder(Class<T>): T
+ revertRecordsFrom(RecordListCheckPoint)
+ createRecordListCheckPoint(): RecordListCheckPoint


<<interface>>
ThrottleContext
+ shouldThrottleNOfUnscaled(int, HederaFunctionality): boolean
+ hasThrottleCapacityForChildTransaction(): boolean
+ getUsageSnapshots(): List<DeterministicThrottle.UsageSnaphot>
+ resetUsageThrottlesTo(List<DeterministicThrottle.UsageSnapshot>


<<interface>>
StakingRewardAccumulator
+ addReward(AccountID, long)