Foundation Class Reference Guide

EsiObjects V4.1

ESI Technology Corporation
211 Vaughn Hill Road
Bolton, MA. 01740
www.esitechnology.com

Table of Contents

FOUNDATION CLASS REFERENCE GUIDE ..... **I**

ESIOBJECTS V4.1 ..... **I**

TABLE OF CONTENTS ..... **III**

CLASS: BASE$ABSATTACHMENTOBJECT **1**

INTERFACE: ATTACHMENT 1
Method: DESTROY *1*
Property: Class *1*
Property: Object *1*
Property: Token *1*

CLASS: BASE$ABSFACTORYOBJECT **1**

INTERFACE: CORE 1
Event: Dead *2*
Method: AddRef *2*
Method: CopyInstanceTable *2*
Method: ReleaseRef *2*
Method: Validate *2*
Property: Class *2*
Property: ClassName *2*
Property: Domain *3*
Property: Id *3*
Property: Name *3*
INTERFACE: FACTORY 3
Event: ObjectDead *3*
Method: CREATE *4*
Method: DESTROY *4*

CLASS: BASE$ABSLOCKABLEOBJECT **4**

INTERFACE: LOCKCONTROL 4
Method: Lock *4*
Method: Unlock *4*
INTERFACE: PRIMARY 4
Method: Lock *5*
Method: Unlock *5*

CLASS: BASE$ABSSECURITYOBJECT **5**

INTERFACE: PRIMARY 5
Method: GetACL *5*
Method: VerifyAccess *5*
INTERFACE: SECURITY 5
Method: Secure *5*

CLASS: BASE$ABSSERIALIZATIONOBJECT **6**

INTERFACE: SERIALIZATION 6
Method: Dump *6*
Method: Export *6*
Method: Import *6*
Method: RestoreFormatted *6*
Method: SaveFormatted *6*

CLASS: BASE$ANDCOMPOUNDCRITERIA **7**

INTERFACE: FACTORY 7
Method: InitClassVars *7*
Method: InitSysVars *7*
INTERFACE: PRIMARY 7
Method: Matches *7*
INSTANCE VARIABLES: 7

CLASS: BASE$ARRAY 7

INTERFACE: FACTORY 8
Method: CREATE *8*
Method: DestroyAll *8*
Method: InitAllSysVars *8*
Method: InitClassVars *8*
INTERFACE: PRIMARY 8
Method: ContainsElement *8*
Method: Copy *8*
Method: CreateIterator *9*
Method: InsertElement *9*
Method: InsertElementAt *9*
Method: RemoveAll *9*
Method: RemoveAllOccurrences *9*
Method: RemoveElement *9*
Method: RemoveElementAt *10*
Method: ReplaceElementAt *10*
Method: Resize *10*
Method: RetrieveElementAt *10*
Property: AllowsDuplicates *10*
Property: IsOrdered *11*
Property: Length *11*
INSTANCE VARIABLES: 11
I%Length *11*

CLASS: BASE$ARRAY>ITERATOR 11

INTERFACE: FACTORY 12
Method: InitAllSysVars *12*
Method: InitClassVars *12*
Method: InitSysVars *12*
INTERFACE: PRIMARY 12
Method: First *12*
Method: Last *12*
Method: More *12*
Method: Next *13*
Property: IterationOrder *13*
INSTANCE VARIABLES: 13

CLASS: BASE$ARRAYITERATOR 13

INTERFACE: FACTORY 13
Method: InitAllSysVars *13*
Method: InitClassVars *13*
Method: InitSysVars *14*
INTERFACE: PRIMARY 14
Method: First *14*
Method: Last *14*
Method: More *14*
Method: Next *14*
Property: IterationOrder *14*

INSTANCE VARIABLES: 15

CLASS: BASE$ATTRIBUTES 15

INTERFACE: PRIMARY 15
Method: getIndex 15
Method: getLength 15
Method: getQName 15
Method: getValue 16

CLASS: BASE$ATTRIBUTESIMPL 16

INTERFACE: PRIMARY 16
Method: addAttribute 16
Method: clear 16
Method: getIndex 16
Method: getLength 17
Method: getLocalName 17
Method: getQName 17
Method: getURI 17
Method: getValue 17
INSTANCE VARIABLES: 17
I%Array 17
I%Length 18

CLASS: BASE$BAG 18

INTERFACE: FACTORY 18
Method: CREATE 18
Method: DestroyAll 18
Method: InitAllSysVars 18
Method: InitClassVars 18
INTERFACE: PRIMARY 19
Method: ContainsElement 19
Method: Copy 19
Method: CreateIterator 19
Method: Difference 19
Method: InsertElement 20
Method: Intersection 20
Method: RemoveAll 20
Method: RemoveAllOccurrences 20
Method: RemoveElement 20
Method: RemoveElementAt 21
Method: ReplaceElementAt 21
Method: RetrieveElementAt 21
Method: Union 21
Property: AllowsDuplicates 22
Property: IsOrdered 22
INTERFACE: VARIABLEFACTORY 22
Method: HasNull 22
INSTANCE VARIABLES: 22
I%HasNull 22
I%IC 22

CLASS: BASE$BAG>ITERATOR 23

INTERFACE: FACTORY 23
Method: CREATE 23
Method: InitAllSysVars 23
Method: InitClassVars 23

Method: InitSysVars *23*
INTERFACE: INTERNAL 23
Method: ElementAdded *24*
Method: ElementRemoved *24*
INTERFACE: PRIMARY 24
Method: Last *24*
Method: Next *24*
INSTANCE VARIABLES: 25
I%HasNull *25*
I%ItemNumber *25*
I%RemoveList *25*
I%SetAsideList *25*
I%Status *25*

CLASS: BASE$BAGITERATOR 26

INTERFACE: FACTORY 26
Method: CREATE *26*
Method: InitAllSysVars *26*
Method: InitClassVars *26*
Method: InitSysVars *26*
INTERFACE: INTERNAL 26
Method: ElementAdded *26*
Method: ElementRemoved *27*
INTERFACE: PRIMARY 27
Method: Last *27*
Method: Next *27*
INSTANCE VARIABLES: 28
I%HasNull *28*
I%ItemNumber *28*
I%RemoveList *28*
I%SetAsideList *28*
I%Status *28*

CLASS: BASE$COLLECTION 28

INTERFACE: FACTORY 29
Method: CREATE *29*
Method: DESTROY *29*
Method: DestroyAll *29*
Method: InitAllSysVars *29*
Method: InitClassVars *29*
Method: InitSysVars *29*
INTERFACE: INTERNAL 30
Method: CopyExternally *30*
Method: GetPointer *30*
Method: RemoveIterator *30*
INTERFACE: PRIMARY 30
Event: AllRemoved *30*
Event: ElementAdded *31*
Event: ElementRemoved *31*
Method: ContainsElement *31*
Method: Copy *31*
Method: CreateIterator *31*
Method: InsertElement *32*
Method: InsertElementAt *32*
Method: KeyType *32*
Method: RemoveAll *32*

Method: RemoveAllOccurrences ... *32*
Method: RemoveElement ... *33*
Method: RemoveElementAt ... *33*
Method: ReplaceElementAt ... *33*
Method: RetrieveElementAt ... *33*
Property: AllowsDuplicates ... *33*
Property: AutoDelete ... *33*
Property: AutoDestroy ... *34*
Property: Capabilities ... *34*
Property: Cardinality ... *35*
Property: IsEmpty ... *35*
Property: IsKeyed ... *35*
Property: IsOrdered ... *35*
Property: Type ... *36*
INSTANCE VARIABLES: ... 36
I%AutoDelete ... *36*
I%AutoDestroy ... *36*
I%Cardinality ... *36*
I%IL ... *36*
I%IteratorList ... *37*
I%Type ... *37*

CLASS: BASE$COLLECTION>ITERATOR ... 37

INTERFACE: FACTORY ... 37
Method: CREATE ... *37*
Method: DESTROY ... *38*
Method: InitAllSysVars ... *38*
Method: InitClassVars ... *38*
Method: InitSysVars ... *38*
INTERFACE: INTERNAL ... 38
Method: ElementAdded ... *38*
Method: ElementRemoved ... *38*
Property: CurrentPosition ... *39*
INTERFACE: PRIMARY ... 39
Method: First ... *39*
Method: Last ... *39*
Method: More ... *39*
Method: Next ... *40*
Method: Reset ... *40*
Property: IterationOrder ... *40*
INSTANCE VARIABLES: ... 40
I%AddList ... *40*
I%CurrentCell ... *41*
I%CurrentItem ... *41*
I%IOrder ... *41*
I%ItemNumber ... *41*
I%List ... *41*
I%OrderKy ... *42*
I%Owner ... *42*
I%RemoveList ... *42*
I%SetAsideList ... *42*
I%Status ... *42*
I%StopNumber ... *43*

CLASS: BASE$COLLECTIONPROTECTOR ... 43

INTERFACE: FACTORY ... 43

Method: CREATE *43*
Method: DESTROY *43*
Method: InitAllSysVars *43*
INTERFACE: PRIMARY 43
Method: ContainsElement *43*
Method: ContainsKey *44*
Method: Copy *44*
Method: CreateIterator *44*
Method: InsertElement *44*
Method: Lock *44*
Method: RemoveAll *44*
Method: RemoveElement *45*
Method: RemoveElementAt *45*
Method: ReplaceElementAt *45*
Method: RetrieveElementAt *45*
Method: RetrieveElementsByKey *45*
Method: RetrieveElementsByPattern *45*
Method: RetrieveKey *45*
Method: Unlock *46*
Property: AllowsDuplicates *46*
Property: AutoDelete *46*
Property: Capabilities *46*
Property: Cardinality *47*
Property: IsEmpty *47*
Property: IsOrdered *47*
Property: Key *48*
INSTANCE VARIABLES: 48

CLASS: BASE$COMPOUNDCRITERIA 48

INTERFACE: FACTORY 48
Method: DESTROY *48*
Method: InitClassVars *48*
Method: InitSysVars *48*
INTERFACE: PRIMARY 48
Method: AddCriteria *49*
Method: RemoveCriteria *49*
Property: Properties *49*
INSTANCE VARIABLES: 49
I%CriteriaList *49*

CLASS: BASE$CONTAINSCRITERIA 49

INTERFACE: FACTORY 49
Method: InitSysVars *50*
INTERFACE: PRIMARY 50
Method: Matches *50*
INSTANCE VARIABLES: 50

CLASS: BASE$CONTENTHANDLER 50

CLASS: BASE$CRITERIA 50

INTERFACE: FACTORY 50
Method: DESTROY *50*
Method: InitClassVars *51*
Method: InitSysVars *51*
INTERFACE: PRIMARY 51
Method: Matches *51*

Property: IsRange *51*
Property: Properties *51*

CLASS: BASE$DATAMANAGER **52**

INTERFACE: FACTORY 52
Method: DESTROY *52*
Method: InitAllSysVars *52*
Method: InitClassVars *52*
Method: InitSysVars *52*
INTERFACE: INTERNAL 52
Method: Watch *53*
INTERFACE: PRIMARY 53
Method: AddKey *53*
Method: ContainsElement *53*
Method: CreateElement *53*
Method: CreateIteratorForKey *53*
Method: InsertElement *54*
Method: RemoveElement *54*
Method: RemoveKey *54*
Method: RetrieveElementsByKey *54*
Method: SelectMatches *54*
Property: Cardinality *55*
Property: Class *55*
Property: ControlsData *55*
Property: InitialKey *56*
Property: Keys *56*
INSTANCE VARIABLES: 56
I%BaseClass *56*
I%BaseList *56*
I%DictionaryList *57*
I%InstanceControl *57*

CLASS: BASE$DATE **57**

INTERFACE: FACTORY 57
Method: CREATE *57*
Method: InitClassVars *58*
Method: InitSysVars *58*
INTERFACE: PRIMARY 58
Method: Add *58*
Method: DaysInMonth *58*
Method: DaysInYear *58*
Method: Decrement *59*
Method: FromString *59*
Method: GreaterThan *59*
Method: GreaterThanOrEqual *59*
Method: Increment *59*
Method: IsBetween *60*
Method: IsEqual *60*
Method: IsValid *60*
Method: IsValidDate *60*
Method: LessThan *60*
Method: LessThanOrEqual *61*
Method: Next *61*
Method: NotEqual *61*
Method: Overlaps *61*
Method: Previous *61*

Method: Subtract ... *62*
Property: Current ... *62*
Property: Day ... *62*
Property: DayOfWeek ... *62*
Property: DayOfYear ... *62*
Property: IsLeapYear ... *63*
Property: Mdate ... *63*
Property: Month ... *63*
Property: TextDate ... *63*
Property: TextMonth ... *63*
Property: Today ... *64*
Property: Year ... *64*

CLASS: BASE$DESCRIPTORS ... 64

CLASS: BASE$DICTIONARY ... 64

INTERFACE: FACTORY ... 65
Method: CREATE ... *65*
Method: DESTROY ... *65*
Method: DestroyAll ... *65*
Method: InitAllSysVars ... *65*
Property: Key ... *65*
INTERFACE: INTERNAL ... 66
Method: KeyModified ... *66*
Method: Watch ... *66*
INTERFACE: PRIMARY ... 66
Method: ContainsElement ... *66*
Method: ContainsKey ... *66*
Method: Copy ... *67*
Method: CreateIterator ... *67*
Method: InsertElement ... *67*
Method: KeyType ... *67*
Method: RemoveAll ... *67*
Method: RemoveElement ... *67*
Method: RemoveElementAt ... *68*
Method: RetrieveElementAt ... *68*
Method: RetrieveElementsByKey ... *68*
Method: RetrieveElementsByPattern ... *68*
Method: RetrieveKey ... *68*
Property: AllowsDuplicates ... *68*
Property: IsOrdered ... *69*
Property: Key ... *69*
INSTANCE VARIABLES: ... 69
I%Key ... *69*
I%NIL ... *70*
I%XIL ... *70*

CLASS: BASE$DICTIONARY>ITERATOR ... 70

INTERFACE: FACTORY ... 70
Method: CREATE ... *70*
Method: InitAllSysVars ... *71*
Method: InitClassVars ... *71*
Method: InitSysVars ... *71*
INTERFACE: INTERNAL ... 71
Method: ElementAdded ... *71*
Method: ElementRemoved ... *71*

Property: CurrentPosition *72*
INTERFACE: PRIMARY 72
Method: First *72*
Method: Last *72*
Method: More *72*
Method: Next *72*
Method: Reset *72*
Property: CurrentKey *73*
Property: EndKey *73*
Property: IterationOrder *73*
Property: StartKey *74*
INSTANCE VARIABLES: 74
I%CurrentKey *74*
I%EndKey *74*
I%StartKey *74*

CLASS: BASE$DICTIONARYITERATOR 75

INTERFACE: FACTORY 75
Method: CREATE *75*
Method: InitAllSysVars *75*
Method: InitClassVars *75*
Method: InitSysVars *75*
INTERFACE: INTERNAL 75
Method: ElementAdded *75*
Method: ElementRemoved *76*
Property: CurrentPosition *76*
INTERFACE: PRIMARY 76
Method: First *76*
Method: Last *76*
Method: More *77*
Method: Next *77*
Method: Reset *77*
Property: CurrentKey *77*
Property: EndKey *77*
Property: IterationOrder *78*
Property: StartKey *78*
INSTANCE VARIABLES: 78
I%CurrentKey *78*
I%EndKey *79*
I%StartKey *79*

CLASS: BASE$DOCBUILDER 79

INTERFACE: FACTORY 79
Method: InitAllSysVars *79*
INTERFACE: PRIMARY 79
Method: GenerateDoc *79*
INSTANCE VARIABLES: 80

CLASS: BASE$EXACTHITCRITERIA 80

INTERFACE: FACTORY 80
Method: CREATE *80*
Method: DESTROY *80*
Method: InitClassVars *80*
Method: InitSysVars *81*
INTERFACE: PRIMARY 81
Method: Matches *81*

Property: Properties *81*
Property: Property *81*
Property: Value *82*
INSTANCE VARIABLES: 82
I%MatchProperty *82*
I%MatchValue *82*

CLASS: BASE$FILTERCRITERIA 82

INTERFACE: FACTORY 82
Method: InitClassVars *83*
Method: InitSysVars *83*
INTERFACE: PRIMARY 83

CLASS: BASE$GREATERTHANCRITERIA 83

INTERFACE: FACTORY 83
Method: InitSysVars *83*
INTERFACE: PRIMARY 83
Method: Matches *83*
INSTANCE VARIABLES: 84

CLASS: BASE$IMMUTABLE 84

INTERFACE: FACTORY 84
Method: InitClassVars *84*
Method: InitSysVars *84*

CLASS: BASE$INTERNALSTREAM 84

INTERFACE: FACTORY 84
Method: InitAllSysVars *85*
Method: InitSysVars *85*
INTERFACE: PRIMARY 85
Method: Read *85*
Method: ReadLine *85*
Method: Reset *85*
Method: Write *85*
Method: WriteLine *85*
INSTANCE VARIABLES: 86
I%Blocks *86*
I%CurBlock *86*
I%CurChar *86*
I%D *86*
I%LastLen *86*
I%Length *87*

CLASS: BASE$INTERVAL 87

INTERFACE: FACTORY 87
Method: CREATE *87*
Method: InitClassVars *88*
Method: InitSysVars *88*
INTERFACE: PRIMARY 88
Method: Add *88*
Method: Divide *88*
Method: FromString *88*
Method: GreaterThan *89*
Method: GreaterThanOrEqual *89*
Method: IsEqual *89*
Method: LessThan *89*

Method: LessThanOrEqual *89*
Method: Mod *89*
Method: Multiply *90*
Method: NotEqual *90*
Method: Subtract *90*
Property: Abs *90*
Property: Day *90*
Property: Hour *91*
Property: IsZero *91*
Property: Minute *91*
Property: Second *91*
Property: TotalHours *92*
Property: TotalMinutes *92*
Property: TotalSeconds *92*

CLASS: BASE$ITERATOR 92

INTERFACE: FACTORY 93
Method: CREATE *93*
Method: DESTROY *93*
Method: InitAllSysVars *93*
Method: InitClassVars *93*
Method: InitSysVars *93*
INTERFACE: INTERNAL 93
Method: ElementAdded *93*
Method: ElementRemoved *94*
Property: CurrentPosition *94*
INTERFACE: PRIMARY 94
Method: First *94*
Method: Last *95*
Method: More *95*
Method: Next *95*
Method: Reset *95*
Property: IterationOrder *95*
INSTANCE VARIABLES: 96
I%AddList *96*
I%CurrentCell *96*
I%CurrentItem *96*
I%IOrder *96*
I%ItemNumber *96*
I%List *97*
I%OrderKy *97*
I%Owner *97*
I%RemoveList *97*
I%SetAsideList *97*
I%Status *98*
I%StopNumber *98*

CLASS: BASE$KEYEDCOLLECTION 98

INTERFACE: FACTORY 98
Method: InitAllSysVars *98*
INTERFACE: PRIMARY 98
Method: ContainsKey *98*
Method: KeyType *99*
Method: RetrieveElementsByKey *99*
Method: RetrieveElementsByPattern *99*
Method: RetrieveKeysByPattern *99*

Property: IsKeyed *99*
INSTANCE VARIABLES: 100

CLASS: BASE$KEYEDCOLLECTION>ITERATOR 100

INTERFACE: FACTORY 100
Method: InitAllSysVars *100*
INTERFACE: PRIMARY 100
Property: CurrentItem *100*
Property: CurrentKey *100*
Property: EndKey *101*
Property: StartKey *101*
INSTANCE VARIABLES: 101

CLASS: BASE$KEYEDITERATOR 101

INTERFACE: FACTORY 102
Method: InitAllSysVars *102*
INTERFACE: PRIMARY 102
Property: CurrentItem *102*
Property: CurrentKey *102*
Property: EndKey *102*
Property: StartKey *103*
INSTANCE VARIABLES: 103

CLASS: BASE$LESSTHANCRITERIA 103

INTERFACE: FACTORY 103
Method: InitSysVars *103*
INTERFACE: PRIMARY 104
Method: Matches *104*
INSTANCE VARIABLES: 104

CLASS: BASE$LIST 104

INTERFACE: FACTORY 104
Method: CREATE *104*
Method: DestroyAll *104*
Method: InitAllSysVars *105*
INTERFACE: PRIMARY 105
Method: ContainsElement *105*
Method: Copy *105*
Method: CreateIterator *105*
Method: InsertElement *105*
Method: InsertElementAfter *105*
Method: InsertElementBefore *106*
Method: InsertFirstElement *106*
Method: InsertLastElement *106*
Method: RemoveAll *106*
Method: RemoveAllOccurrences *106*
Method: RemoveElement *107*
Method: RemoveElementAt *107*
Method: RemoveFirstElement *107*
Method: RemoveLastElement *107*
Method: ReplaceElementAt *107*
Method: RetrieveElementAt *107*
Method: RetrieveFirstElement *108*
Method: RetrieveLastElement *108*
Property: AllowsDuplicates *108*
Property: CurrentPosition *108*

Property: IsOrdered *108*
INSTANCE VARIABLES: 109
I%CurrentPosId *109*
I%CurrentPosition *109*
I%Head *109*
I%Tail *110*

CLASS: BASE$LIST>ITERATOR 110

INTERFACE: FACTORY 110
Method: InitAllSysVars *110*
INTERFACE: INTERNAL 110
Method: ElementAdded *110*
Method: ElementRemoved *111*
INTERFACE: PRIMARY 111
Method: First *111*
Method: Last *111*
Method: More *111*
Method: Next *112*
Property: IterationOrder *112*
INSTANCE VARIABLES: 112

CLASS: BASE$LISTITERATOR 112

INTERFACE: FACTORY 112
Method: InitAllSysVars *112*
INTERFACE: INTERNAL 112
Method: ElementAdded *113*
Method: ElementRemoved *113*
INTERFACE: PRIMARY 113
Method: First *113*
Method: Last *114*
Method: More *114*
Method: Next *114*
Property: IterationOrder *114*
INSTANCE VARIABLES: 114

CLASS: BASE$LOG 114

INTERFACE: FACTORY 115
Method: CREATE *115*
Method: DestroyAll *115*
Method: InitAllSysVars *115*
Method: InitClassVars *115*
INTERFACE: PRIMARY 115
Method: ContainsElement *115*
Method: Copy *115*
Method: CreateIterator *116*
Method: InsertElement *116*
Method: InsertElementAt *116*
Method: RemoveAll *116*
Method: RemoveAllOccurrences *116*
Method: RemoveElement *117*
Method: RemoveElementAt *117*
Method: RetrieveElementAt *117*
Method: RetrieveTimeStamp *117*
Property: AllowsDuplicates *117*
Property: IsOrdered *117*
INSTANCE VARIABLES: 118

CLASS: BASE$LOG>ITERATOR 118

INTERFACE: FACTORY 118
Method: CREATE 118
Method: InitAllSysVars 118
Method: InitClassVars 118
Method: InitSysVars 118
INTERFACE: INTERNAL 119
Method: ElementAdded 119
Method: ElementRemoved 119
Property: CurrentPosition 119
INTERFACE: PRIMARY 120
Method: First 120
Method: Last 120
Method: More 120
Method: MoveTo 120
Method: Next 120
Method: Reset 121
Property: Current 121
Property: IterationOrder 121
INSTANCE VARIABLES: 121
I%LogC 121
I%LogD 121
I%LogS 122

CLASS: BASE$LOGITERATOR 122

INTERFACE: FACTORY 122
Method: CREATE 122
Method: InitAllSysVars 122
Method: InitClassVars 122
Method: InitSysVars 122
INTERFACE: INTERNAL 123
Method: ElementAdded 123
Method: ElementRemoved 123
Property: CurrentPosition 123
INTERFACE: PRIMARY 124
Method: First 124
Method: Last 124
Method: More 124
Method: MoveTo 124
Method: Next 124
Method: Reset 125
Property: Current 125
Property: IterationOrder 125
INSTANCE VARIABLES: 125
I%LogC 125
I%LogD 125
I%LogS 126

CLASS: BASE$MVARIABLE 126

INTERFACE: FACTORY 126
Method: CREATE 126
Method: InitClassVars 126
Method: InitSysVars 126
INTERFACE: PRIMARY 126
Method: Copy 126

Method: Delete ... *127*
Method: Kill ... *127*
Method: Merge ... *127*
Method: Move ... *127*
Property: Data ... *127*
Property: GLVN ... *127*
Property: Value ... *128*

CLASS: BASE$MVARIABLESTREAM ... 128

INTERFACE: FACTORY ... 128
Method: CREATE ... *128*
Method: InitAllSysVars ... *128*
Method: InitSysVars ... *128*
INTERFACE: PRIMARY ... 129
Method: Read ... *129*
Method: ReadLine ... *129*
Method: Reset ... *129*
Method: Write ... *129*
Method: WriteLine ... *129*
INSTANCE VARIABLES: ... 129
I%Blocks ... *129*
I%CurBlock ... *130*
I%CurChar ... *130*
I%LastLen ... *130*
I%Length ... *130*
I%Root ... *130*

CLASS: BASE$MAP ... 130

INTERFACE: FACTORY ... 131
Method: CREATE ... *131*
Method: DESTROY ... *131*
Method: DestroyAll ... *131*
Method: InitAllSysVars ... *131*
Method: InitClassVars ... *131*
INTERFACE: PRIMARY ... 131
Event: ElementAdded ... *131*
Event: ElementRemoved ... *132*
Method: ContainsElement ... *132*
Method: ContainsKey ... *132*
Method: Copy ... *132*
Method: CreateIterator ... *132*
Method: DestroyAll ... *133*
Method: InsertElement ... *133*
Method: InsertElementAt ... *133*
Method: Merge ... *133*
Method: RemoveAll ... *133*
Method: RemoveElement ... *134*
Method: RemoveElementAt ... *134*
Method: ReplaceElementAt ... *134*
Method: RetrieveElement ... *134*
Method: RetrieveElementAt ... *134*
Method: RetrieveElementsByKey ... *135*
Method: RetrieveElementsByPattern ... *135*
Method: SetElement ... *135*
Property: AllowsDuplicates ... *135*
Property: IsOrdered ... *135*

INSTANCE VARIABLES: 136
I%IL 136
I%NIL 136

CLASS: BASE$MAP>ITERATOR 136

INTERFACE: FACTORY 136
Method: CREATE 137
Method: DESTROY 137
Method: InitAllSysVars 137
Method: InitClassVars 137
Method: InitSysVars 137
INTERFACE: INTERNAL 137
Method: ElementAdded 137
Method: ElementRemoved 138
Method: SuperSort 138
Property: CurrentPosition 138
INTERFACE: PRIMARY 139
Method: First 139
Method: Last 139
Method: More 139
Method: Next 139
Method: Reset 139
Property: CurrentItem 140
Property: CurrentKey 140
Property: DeadSpot 140
Property: EndKey 141
Property: IterationOrder 141
Property: StartKey 141
INSTANCE VARIABLES: 142
I%CurrentKey 142
I%DeadSpot 142
I%EndKey 142
I%NumOrder 142
I%StartKey 142

CLASS: BASE$MAPITERATOR 143

INTERFACE: FACTORY 143
Method: CREATE 143
Method: DESTROY 143
Method: InitAllSysVars 143
Method: InitClassVars 143
Method: InitSysVars 144
INTERFACE: INTERNAL 144
Method: ElementAdded 144
Method: ElementRemoved 144
Method: SuperSort 144
Property: CurrentPosition 145
INTERFACE: PRIMARY 145
Method: First 145
Method: Last 145
Method: More 145
Method: Next 146
Method: Reset 146
Property: CurrentItem 146
Property: CurrentKey 146
Property: DeadSpot 146

Property: EndKey *147*
Property: IterationOrder *147*
Property: StartKey *148*
INSTANCE VARIABLES: 148
I%CurrentKey *148*
I%DeadSpot *148*
I%EndKey *148*
I%NumOrder *148*
I%StartKey *149*

CLASS: BASE$MULTIMAP 149

INTERFACE: FACTORY 149
Method: CREATE *149*
Method: DestroyAll *149*
Method: InitAllSysVars *150*
Method: InitClassVars *150*
INTERFACE: INTERNAL 150
INTERFACE: PRIMARY 150
Event: ElementAdded *150*
Event: ElementRemoved *150*
Method: ContainsElement *150*
Method: ContainsKey *151*
Method: Copy *151*
Method: CreateIterator *151*
Method: CreateKeyIterator *151*
Method: InsertElement *151*
Method: RemoveAll *151*
Method: RemoveElement *152*
Method: RemoveElementAt *152*
Method: ReplaceElementAt *152*
Method: RetrieveElement *152*
Method: RetrieveElementAt *152*
Method: RetrieveElementsByKey *152*
Method: RetrieveElementsByPattern *153*
Method: RetrieveKeysByPattern *153*
Property: AllowsDuplicates *153*
Property: IsOrdered *153*
INSTANCE VARIABLES: 154
I%IL *154*
I%NIL *154*

CLASS: BASE$MULTIMAP>ITERATOR 154

INTERFACE: FACTORY 154
Method: CREATE *155*
Method: InitAllSysVars *155*
Method: InitClassVars *155*
Method: InitSysVars *155*
INTERFACE: INTERNAL 155
Method: ElementRemoved *155*
INTERFACE: PRIMARY 155
Method: First *156*
Method: Last *156*
Method: Next *156*
Property: CurrentItem *156*
Property: CurrentKey *156*
Property: DeadSpot *157*

Property: IterationOrder *157*
INSTANCE VARIABLES: 158
I%CurrentKey *158*
I%DeadSpot *158*
I%EndKey *158*
I%MarkerOne *158*
I%MarkerTwo *158*
I%NumOrder *159*
I%StartKey *159*

CLASS: BASE$MULTIMAPITERATOR 159

INTERFACE: FACTORY 159
Method: CREATE *159*
Method: InitAllSysVars *159*
Method: InitClassVars *159*
Method: InitSysVars *160*
INTERFACE: INTERNAL 160
Method: ElementRemoved *160*
INTERFACE: PRIMARY 160
Method: First *160*
Method: Last *160*
Method: Next *161*
Property: CurrentItem *161*
Property: CurrentKey *161*
Property: DeadSpot *161*
Property: IterationOrder *162*
INSTANCE VARIABLES: 162
I%CurrentKey *162*
I%DeadSpot *162*
I%EndKey *163*
I%MarkerOne *163*
I%MarkerTwo *163*
I%NumOrder *163*
I%StartKey *163*

CLASS: BASE$MULTIMAPKEYITERATOR 164

INTERFACE: FACTORY 164
Method: InitAllSysVars *164*
INTERFACE: PRIMARY 164
Method: First *164*
Method: Last *164*
Method: More *164*
Method: Next *165*
INSTANCE VARIABLES: 165

CLASS: BASE$NAMEVALUEPAIR 165

INTERFACE: FACTORY 165
Method: CREATE *165*
Method: InitClassVars *165*
Method: InitSysVars *165*
INTERFACE: PRIMARY 165
Property: Name *165*
Property: Value *166*

CLASS: BASE$NEWNAMEPOOL 166

INTERFACE: FACTORY 166

Method: CREATE *166*
Method: CreateDescendant *167*
Method: DESTROY *167*
Method: DestroySymbols *167*
Method: InitClassVars *167*
Method: InitSysVars *167*
INTERFACE: PRIMARY 167
Method: AncestorLink *167*
Method: AncestorUnlink *168*
Method: DescendantLink *168*
Method: DescendantUnlink *168*
Method: ReportLink *168*
Property: Name *168*
INTERFACE: VARIABLEFACTORY 169
Method: Descendants *169*
Method: Name *169*
INSTANCE VARIABLES: 169
I%Ancestor *169*
I%Descendants *169*
I%Name *169*

CLASS: BASE$OLEOBJECT 170

INTERFACE: FACTORY 170
Method: CREATE *170*
INTERFACE: PRIMARY 170
Method: Test *170*

CLASS: BASE$ORCOMPOUNDCRITERIA 170

INTERFACE: FACTORY 170
Method: InitClassVars *170*
Method: InitSysVars *170*
INTERFACE: PRIMARY 170
Method: Matches *171*
INSTANCE VARIABLES: 171

CLASS: BASE$PATTERNCRITERIA 171

INTERFACE: FACTORY 171
Method: CREATE *171*
Method: DESTROY *171*
Method: InitClassVars *171*
Method: InitSysVars *171*
INTERFACE: PRIMARY 171
Method: Matches *172*
Property: Pattern *172*
Property: Properties *172*
Property: Property *172*
INSTANCE VARIABLES: 173
I%PatProperty *173*
I%Pattern *173*

CLASS: BASE$RANGECRITERIA 173

INTERFACE: FACTORY 173
Method: CREATE *173*
Method: DESTROY *173*
Method: InitClassVars *174*
Method: InitSysVars *174*

INTERFACE: PRIMARY 174
Method: Matches *174*
Property: IsRange *174*
Property: Properties *174*
Property: Property *175*
Property: RangeEnd *175*
Property: RangeStart *175*
INSTANCE VARIABLES: 175
I%RngEnd *175*
I%RngProperty *176*
I%RngStart *176*

CLASS: BASE$READER 176

CLASS: BASE$RELATIONALCRITERIA 176

INTERFACE: FACTORY 176
Method: CREATE *176*
Method: DESTROY *176*
Method: InitSysVars *177*
INTERFACE: PRIMARY 177
Property: Properties *177*
Property: Property *177*
Property: Value *177*
INSTANCE VARIABLES: 178
I%Property *178*
I%RelationalValue *178*

CLASS: BASE$SERVERFILESTREAM 178

INTERFACE: FACTORY 178
Method: CREATE *178*
Method: DESTROY *178*
Method: InitAllSysVars *179*
Method: InitSysVars *179*
INTERFACE: PRIMARY 179
Method: Close *179*
Method: Open *179*
Method: Read *179*
Method: ReadLine *179*
Method: Write *179*
Method: WriteLine *179*
INSTANCE VARIABLES: 180
I%File *180*
I%LastChar *180*
I%Mode *180*

CLASS: BASE$SET 180

INTERFACE: FACTORY 180
Method: CREATE *180*
Method: DestroyAll *181*
Method: InitAllSysVars *181*
Method: InitClassVars *181*
INTERFACE: PRIMARY 181
Method: ContainsElement *181*
Method: Copy *181*
Method: CreateIterator *181*
Method: Difference *182*

Method: InsertElement *182*
Method: Intersection *182*
Method: IsProperSubset *182*
Method: IsProperSuperset *183*
Method: IsSubset *183*
Method: IsSuperset *183*
Method: RemoveAll *183*
Method: RemoveElement *183*
Method: RemoveElementAt *184*
Method: ReplaceElementAt *184*
Method: RetrieveElementAt *184*
Method: Union *184*
Property: AllowsDuplicates *184*
Property: IsOrdered *185*
INTERFACE: VARIABLEFACTORY 185
Method: HasNull *185*
INSTANCE VARIABLES: 185
I%HasNull *185*
I%Items *185*

CLASS: BASE$SET>ITERATOR 186

INTERFACE: FACTORY 186
Method: CREATE *186*
Method: InitAllSysVars *186*
Method: InitClassVars *186*
Method: InitSysVars *186*
INTERFACE: INTERNAL 186
Method: ElementAdded *186*
Method: ElementRemoved *187*
INTERFACE: PRIMARY 187
Method: Last *187*
Method: Next *187*
INSTANCE VARIABLES: 188
I%HasNull *188*

CLASS: BASE$SETITERATOR 188

INTERFACE: FACTORY 188
Method: CREATE *188*
Method: InitAllSysVars *188*
Method: InitClassVars *188*
Method: InitSysVars *188*
INTERFACE: INTERNAL 188
Method: ElementAdded *189*
Method: ElementRemoved *189*
INTERFACE: PRIMARY 189
Method: Last *189*
Method: Next *189*
INSTANCE VARIABLES: 190
I%HasNull *190*

CLASS: BASE$SORTSAFTERCRITERIA 190

INTERFACE: FACTORY 190
Method: InitSysVars *190*
INTERFACE: PRIMARY 190
Method: Matches *190*
INSTANCE VARIABLES: 190

CLASS: BASE$STREAM 190

INTERFACE: FACTORY 191
Method: InitAllSysVars 191
Method: InitSysVars 191
INTERFACE: PRIMARY 191
Method: Commit 191
Method: Read 191
Method: ReadLine 191
Method: Reset 191
Method: Write 191
Method: WriteLine 192
INSTANCE VARIABLES: 192
I%ErrorId 192
I%ErrorText 192
I%ProxyType 192

CLASS: BASE$TEXT 192

INTERFACE: FACTORY 193
Method: InitAllSysVars 193
Method: InitClassVars 193
Method: InitSysVars 193
INTERFACE: INTERNAL 193
Method: CreateLineMap 193
Method: Dump 193
INTERFACE: PRIMARY 193
Method: Append 193
Method: AppendLine 194
Method: AppendTextObject 194
Method: BlockCount 194
Method: Blocks 194
Method: Clear 194
Method: Copy 194
Method: GetBlock 194
Method: GetDimension 195
Method: GetLine 195
Method: GetStream 195
Method: GetSubstring 195
Method: GetText 195
Method: HasContent 195
Method: IsEmpty 195
Method: SetDimension 196
Method: SetText 196
Property: LineCount 196
INSTANCE VARIABLES: 196
I%Blocks 196
I%LastLen 196
I%Length 196
I%LineMap 197
I%Lines 197
I%MapValid 197
I%ProxyType 197
I%Text 197
I%Valid 198

CLASS: BASE$TEXT>STREAM 198

INTERFACE: FACTORY 198
Method: CREATE *198*
Method: InitAllSysVars *198*
Method: InitSysVars *198*
INTERFACE: INTERNAL 198
Method: UpdateInfo *199*
INTERFACE: PRIMARY 199
Method: Commit *199*
Method: Read *199*
Method: ReadLine *199*
Method: Reset *199*
Method: Write *199*
Method: WriteLine *199*
INSTANCE VARIABLES: 200
I%Blocks *200*
I%CurBlock *200*
I%CurChar *200*
I%LastLen *200*
I%Length *200*
I%Parent *201*
I%Root *201*
I%Valid *201*
I%VectLength *201*

CLASS: BASE$TIME 201

INTERFACE: FACTORY 202
Method: CREATE *202*
Method: InitClassVars *202*
Method: InitSysVars *202*
INTERFACE: PRIMARY 202
Method: Add *202*
Method: FromString *202*
Method: GreaterThan *203*
Method: GreaterThanOrEqual *203*
Method: IsEqual *203*
Method: IsValid *203*
Method: LessThan *203*
Method: LessThanOrEqual *203*
Method: NotEqual *204*
Method: Subtract *204*
Property: Current *204*
Property: Hour *204*
Property: Minute *204*
Property: Second *205*

CLASS: BASE$TIMERANGE 205

INTERFACE: FACTORY 205
Method: CREATE *205*
Method: InitClassVars *205*
Method: InitSysVars *205*
INTERFACE: PRIMARY 206
Method: IsWithin *206*
Method: Overlap *206*
Property: Duration *206*
Property: End *206*
Property: Start *206*

CLASS: BASE$TIMESTAMP 207

INTERFACE: FACTORY 207
Method: CREATE 207
Method: InitClassVars 207
Method: InitSysVars 207
INTERFACE: PRIMARY 207
Method: Add 207
Method: Date 208
Method: FromString 208
Method: GreaterThan 208
Method: GreaterThanOrEqual 208
Method: IsBetween 208
Method: IsEqual 208
Method: LessThan 209
Method: LessThanOrEqual 209
Method: NotEqual 209
Method: Subtract 209
Method: Time 209
Property: COASFormat 209
Property: CORBAFormat 210
Property: Current 211
Property: Day 211
Property: FileManFormat 211
Property: Hour 211
Property: Minute 212
Property: Month 212
Property: Mstamp 212
Property: OQLFormat 212
Property: Second 213
Property: SortNumber 213
Property: Year 213

CLASS: BASE$TOOLS 213

INTERFACE: FACTORY 214
Method: InitAllSysVars 214
Method: InitSysVars 214
INSTANCE VARIABLES: 214
I%FindCriteria 214
I%Visitor 214

CLASS: BASE$VERSIONED 214

INTERFACE: VERSION 215
Method: CheckUpgrade 215
Method: CompareVersions 215
Method: Upgrade 215
Property: Version 215
INSTANCE VARIABLES: 215
I%Version 215

CLASS: BASE$XMLCONTENTHANDLER 216

INTERFACE: PRIMARY 216
Method: characters 216
Method: endDocument 216
Method: endElement 216
Method: processingInstruction 217

Method: startDocument *217*
Method: startElement *217*
INTERFACE: UTILITY 217
Method: Normalize *217*
Method: ToText *218*

CLASS: BASE$XMLREADER 218

INTERFACE: PRIMARY 218
Method: parse *218*
Method: setContentHandler *218*
Method: setErrorHandler *218*
INSTANCE VARIABLES: 219
I%ContentHandler *219*
I%ErrorHandler *219*

CLASS: BASE$XMLREADERIMPL 219

INTERFACE: INTERNAL 219
Method: Attribute *219*
Method: CharRef *219*
Method: DoctypeDecl *220*
Method: EncodingDecl *220*
Method: EntityRef *220*
Method: Misc *220*
Method: SDDecl *220*
Method: SpecialAttribute *220*
Method: VersionInfo *221*
Method: XMLDecl *221*
Method: content *221*
Method: document *221*
Method: element *221*
Method: main *222*
Method: prolog *222*
INTERFACE: PRIMARY 222
Method: ShowError *222*
Method: getError *222*
Method: parse *222*
INSTANCE VARIABLES: 223
I%Attrs *223*
I%CharData *223*
I%Entities *223*
I%ErrLine *223*
I%Error *223*
I%NamespaceMgr *224*
I%Namespaces *224*
I%Stream *224*

CLASS: BASE$XMLREADERIMPL>NAMESPACEMGR 224

INTERFACE: PRIMARY 224
Method: Define *224*
Method: Lookup *225*
Method: LookupDefault *225*
Method: Pop *225*
Method: Push *225*
Property: LocalDefine *225*
INSTANCE VARIABLES: 225
I%Default *225*

I%Level *226*
I%URI *226*

Class: Base$AbsAttachmentObject

Description: This interface allows other objects to attach data to the object

Type: Mix-In

Interface: Attachment

Description: This interface containss the attachement services.

Method: DESTROY

Description: An abstract template for the DESTROY method.

Options: Public,Method,Void,Abstract
Parameters: None

Property: Class

Description: An abstract template for Class property accessors.

Options: Public,Inheritable

Property: Object

Description: An abstract template for the Object property accessors.

Options: Public,Inheritable

Property: Token

Description: An abstract template for the Token property accessors.

Options: Public,Inheritable

Class: Base$AbsFactoryObject

Description: The factory interface may assist in the life-cycle services needed to create, maintain and destroy objects of the class that inherit the interface. This class is used to add the factory interface to your class. It adds interfaces Factory and Core.

Type: Mix-In

Interface: Core

Description: Contains the core operations for the AbsFactoryObject target class.

Event: Dead

Description: An Event template for implementing an event on and object when it is destroyed.

Callback Documentation:

Method: AddRef

Description: An abstract method template for adding a reference.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: CopyInstanceTable

Description: Copies the current objects instance table into an array of a specified object. Requires privilege to perform this operation.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Object,In)To:T%To

Method: ReleaseRef

Description: A template for implementing the method responsible for releasing a reference on the object.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: Validate

Description: A template for implementing the Validation method of the target object.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: Class

Description: Provides accessor operations on the class OID.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: ClassName

Description: Provides accessor operations on the target classes name.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: Domain

Description: A template for implementing Domain accessors.

Options: Public,Inheritable

Property: Id

Description: A template for implementing accessor operations on the current objects ID,

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: Name

Description: A template for implementing object Name accessors.

Options: Public,Inheritable

Interface: Factory

Description: Contains the constructor and destructor operations for the AbsFactoryObject target class.

Event: ObjectDead

Description: The ObjectDead Event indicates the object has been destroyed. The system will automatically throw this event.

Callback Documentation:

```
Input:  (
        Object:P%Obj,           ;The Oid of the Object.
        Event:P%Evt   ;Always = "Factory>ObjectDead"
        )
        ;
        ;Remove any references the P%Obj.
        ;Note: $EXIST(P%Obj)=0
```

Method: CREATE

Description: The constructor method is used to create any required structures of the object at creation time. This generally includes all static members, and all relationships.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: DESTROY

Description: This destructor method cleans up an object environment at object destruction time.

Options: Public,Method,Void,Platform=All
Parameters: None

Class: Base$AbsLockableObject

Description: This interface allows locking and unlocking operations to be performed on the objects instance table

Type: Mix-In

Interface: LockControl

Description: The private interface for locking and unlocking the current objects instance table.

Method: Lock

Description: Locks an objects instance table.

Options: Private,Method,Void,Inheritable,UsesIO,Platform=All
Parameters:
(Optional,Type=String,In)Timeout:P%Timeout,
(Optional,Type=String,In)Interactive:P%Interactive=0

Method: Unlock

Description: Unlocks an objects instance table.

Options: Private,Method,Void,Inheritable,Platform=All
Parameters: (Optional,Type=String,In)All:P%All=0

Interface: Primary

Description: The public interface for locking and unlocking an objects instance table. It is here for historical purposes. Use the Lock and UnLock methods in the LockControl interface instead.

Method: Lock

Description: Locks an objects instance table.

Options: Private,Method,Void,Inheritable,UsesIO,Platform=All
Parameters:
(Optional,Type=String,In)Timeout:P%Timeout,
(Optional,Type=String,In)Interactive:P%Interactive=0

Method: Unlock

Description: Unlocks an objects instance table.

Options: Private,Method,Void,Inheritable,Platform=All
Parameters: (Optional,Type=String,In)All:P%All=0

Class: Base$AbsSecurityObject

Description: An abstract mixin class that implements a Security interface for the target class to manipulate the security state of the object and provide information services within the primary interface.

Type: Mix-In

Interface: Primary

Description: Public interface that offers public informational sercurity services.

Method: GetACL

Description: A template that provides access to the access control list.

Options: Public,Method,Void
Parameters: None

Method: VerifyAccess

Description: A template used to validate creation and destruction of secury information.

Options: Public,Method,Void
Parameters: None

Interface: Security

Description: Interface that holds the private services that manipulate the security state of the object.

Method: Secure

Description: Provides the service to change the security state of the object.

Options: Public,Method,Void
Parameters: None

Class: Base$AbsSerializationObject

Description: An abstract mixin class that implements a Serialization interface providing services for the target class to use the serialization devices.

Type: Mix-In

Interface: Serialization

Description: Private interface that offers serialization services.

Method: Dump

Description: A template that presents a serial dump of the object for diagnostic purposes.

Options: Public,Method,Void
Parameters: None

Method: Export

Description: A template for the exporting of the object.

Options: Public,Method,Void
Parameters: None

Method: Import

Description: A template for the importing of the object.

Options: Public,Method,Void
Parameters: None

Method: RestoreFormatted

Description: A template for restoring the object from a serial source in the EsiObjects Object interchange format.

Options: Public,Method,Void
Parameters: None

Method: SaveFormatted

Description: A template for saving the object to a serial source in the EsiObjects Object interchange format.

Options: Public,Method,Void

Parameters: None

Class: Base$AndCompoundCriteria

A conjunctive compound Criteria, returning TRUE when all of its component criteria are true, and FALSE if any of them is not true (or if it does not contain any criteria.)
Type: Concrete
Superclasses: CompoundCriteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Method: Matches

Description: Returns true if the ALL of the specified criteria are true, or false if any of them are false. Also returns false if there are no criteria.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)o:P%Object

Instance Variables:

Class: Base$Array

Description: Arrays are ordered lists of elements where the order is determined by a numeric index. Arrays allow the same operations as the collection parent class with the addition of the Resize method. No order is assumed except the order of the numeric index. Duplicate values in each array "cell" are allowed.

Arrays emulate non-sparse arrays found in other programming languages, in that you can determine the "Length" of the array, ie the number of "cells" in the array, each of which may or may not have an element in them. They are not sparse.

Type: Concrete

Superclasses: Collection

Interface: Factory

Method: CREATE

Description: CREATE builds the instance of the array. Initially the cardinality is zero and the length of the array is set to zero. If data is passed into the array on the create, it is inserted. Each insertion changes both the cardinality and the length of the array.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DestroyAll

Description: The DestroyAll method is an internal method that destroys all elements of the collection. It is used to clear the collection of elements for reuse.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Method: ContainsElement

Description: Determines whether the collection contains a specific element.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)o:P%Item,
(Optional,Type=String,In)Item:P%Item

Method: Copy

Description: Copies the elements of an Array collection into a target collection. The target collection may be any collection type compatible with the Array collection.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)Array:P%Target

Method: CreateIterator

Description: Creates an Array iterator object that is used to traverse an Array collection. Execution of this method is the only way an iterator can be created.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InsertElement

Description: InsertElement inserts a new element into the Array collection.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Item

Method: InsertElementAt

Description: InsertElementAt inserts the data element in the Array collection at a specified position.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=Variant,In)o:P%Item,
(Required,Type=Variant,In)Position:P%Pos

Method: RemoveAll

Description: Removes all the Array items and sets the cardinality and length to 0. If AutoDestroy is true, all objects pointed to will be destroyed as well.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RemoveAllOccurrences

Description : Removes all items from a collection that allows duplicates. This abstract method is overriden in subclasses that allow duplicates. For those collections that do not allow duplicates, this method calls RemoveElement.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)o:P%Item,
(Required,Type=Variant,In)Item:P%Item

Method: RemoveElement

Description: Removes the specified item from the collection. RemoveElement finds the element by its value and removes it. Note, if there are two identical elements in the array, only the first one is removed.

Options: Public,Method,Void,Platform=All

Parameters: (Required,Type=Variant,In)o:P%Item

Method: RemoveElementAt

Description: RemoveElementAt positionally removes an element from the array independent of its value. Either an iteration object or an actual position can be specified.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)P%Unknown="",
(Optional,Type=Base$Array>Iterator,In)CurrentPosition:P%Iter,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ReplaceElementAt

Description: Replaces the element pointed to by the position iterator with the item passed in.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Variant,In)o:P%Item,
(Optional,Type=Variant,In)Unknown:P%Unknown,
(Optional,Type=Numeric,In)Position:P%Pos,
(Optional,Type=Base$Array>Iterator,In)CurrentPosition:P%Iter

Method: Resize

Description: Resize changes the length of the array. If the length is increased, the elements in the array are unaffected. If the length is decreased the elements beyond the new end of the array are removed.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)NewSize:P%Size

Method: RetrieveElementAt

Description: RetrieveElementAt retrieves an element of the array based on it's position.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Variant,In)Unknown:P%Unknown,
(Optional,Type=Base$Array>Iterator,In)CurrentPosition:P%Iter,
(Optional,Type=Numeric,In)Position:P%Pos

Property: AllowsDuplicates

Description: AllowsDuplicates is a read-only property that indicates whether or not duplicates are allowed in the given collection. Arrays allow duplicates.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IsOrdered

Description: IsOrdered returns a bollean value. A 0 is returned If the collection is unordered. A 1 is returned If the collection is ordered. Arrays are always ordered and this property will return a 1.

"Ordered" is defined to reflect the constancy of retrieval. For example, if an iterator is used to retrieve the next element from a specified position in an array, it can be assumed that the same data element will be presented each time the call is made-- provided there are no inserts or removes done between the calls.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Length

Description: Length retruns the number of cells in the Array object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

I%Length

Description: Length contains the number of "cells" in this array. This differs from Cardinality, since Cardinality refers to the number of elements in the array. Cardinality, therefore will never be greater than Length.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Array>Iterator

Description: An ArrayIterator object contains iterator functions that operate specifically on a Array collection. It can only be created through the CreateIterator method in the Primary interface of the Array object.

Type: Concrete
Superclasses: Collection>Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the Array iterator.

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given Array collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the Array collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the Array collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All

Parameters: None

Method: Next

Description: Next returns the next element in the Array collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Order:P%Order

Instance Variables:

Class: Base$ArrayIterator

Description: An ArrayIterator object contains iterator functions that operate specifically on a Array collection. It can only be created through the CreateIterator method in the Primary interface of the Array object.

Type: Concrete
Superclasses: Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the Array iterator.

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given Array collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the Array collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the Array collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Next

Description: Next returns the next element in the Array collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Order:P%Order

Instance Variables:

Class: Base$Attributes

Description: This abstract class corresponds to the SAX2 Attributes interface. The interface is implemented by the concrete AttributesImpl subclass.

Type: Abstract
Superclasses: Descriptors

Interface: Primary

Description: Public interface that defines the abstract Attibutes services for the XML parser.

Method: getIndex

Description: Abstract definition that returns an index between 0 and I%Length-1 corresponding to the inputs, which identify an Attribute name, either a URI/localName pair or a qName. Returns -1 if the inputs do not correspond to an attribute in the Array.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Optional,Type=String,In)Name:P%Name=""

Method: getLength

Description: Abstract template that returns the number of Attributes currently represented in the array.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: getQName

Description: Abstract template that returns the qName (qualified name) for the attribute corresponding to a given index.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)Index:P%Index=0

Method: getValue

Description: Abstract template that returns the value for the attribute corresponding to given inputs.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Optional,Type=Variant,In)Name:P%Name=""

Class: Base$AttributesImpl

Description: This class implements the SAX2 Attributes interface.

Type: Concrete
Superclasses: Attributes

Interface: Primary

Description: Public interface that defines the concrete Attibutes service implementation for the XML parser.

Method: addAttribute

Description: Adds an attribute to the array.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=String,In)uri:P%uri="",
(Required,Type=String,In)LocalName:P%localName="",
(Required,Type=String,In)qName:P%qName="",
(Optional,Type=String,In)P%type,
(Optional,Type=String,In)P%value=""

Method: clear

Description: Removes all attributes from the array.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: getIndex

Description: Returns an index between 0 and I%Length-1 corresponding to the inputs, which identify an Attribute name, either a URI/localName pair or a qName. Returns -1 if the inputs do not correspond to an attribute in the Array.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)Name1:P%Name1="",
(Optional,Type=String,In)Name2:P%Name2=""

Method: getLength

Description: Returns the number of Attributes currently represented in the array.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: getLocalName

Description: Returns the localName for the attribute corresponding to a given index.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)Index:P%Index=0

Method: getQName

Description: Returns the qName (qualified name) for the attribute corresponding to a given index.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)Index:P%Index=0

Method: getURI

Description: Returns the URI for the attribute corresponding to a given index.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)Index:P%Index=0

Method: getValue

Description: Returns the value for the attribute corresponding to given inputs.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Variant,In)Name1:P%Name1="",
(Optional,Type=Variant,In)Name2:P%Name2=""

Instance Variables:

I%Array

Description: Array whose elements correspond to individual attributes. Subscripts defined from 0 through I%Length-1. I%Array(i) consists of four pieces delimited by "^":
(1) URI
(2) localName
(3) qualifiedName
(4) value.

Inheritable: Yes

Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%Length

Description: Number of attributes defined in I%Array. Exposed by getLength method.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Bag

Description: A Bag is an unordered Collection of elements which allows duplicates (simular to the Set collection).

Type: Concrete
Superclasses: Collection

Interface: Factory

Method: CREATE

Description: CREATE creates the bag object. P%Data is a list of element to be inserted. P%Data itself contains the number of entries in the P%Data array. Note that data elements may or may not be passed in. The bag is created regardless.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DestroyAll

Description: The DestroyAll method is an internal method that destroys all elements of the collection. It is used to clear the collection of elements for reuse.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Method: ContainsElement

Description: Determines whether the collection contains a specific element.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)o:P%Item,
(Optional,Type=String,In)Item:P%Item

Method: Copy

Description: Copies the elements of an Bag collection into a target collection. The target collection may be any collection type compatible with the Bag collection.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)Bag:P%Target

Method: CreateIterator

Description: Creates an Bag iterator object that is used to traverse an Bag collection. Execution of this method is the only way an iterator can be created.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Difference

Description: Difference returns a new bag object which contains bag elements that are the difference between the bag being called and the bag being submitted. For example, if bag A has:
    1 onion
    3 carrots
    6 peas
and bag B has:
    1 c
arrot
    4 peas
Then the difference bag would contain:
    1 onion
    2 carrots
    2 peas

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Bag,In)Bag:P%Bag

Method: InsertElement

Description: InsertElement inserts a new element into the Bag collection. No presumption of position can be made as to the location of the element in the Bag.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Item

Method: Intersection

Description: Intersection returns a new bag object which contains bag elements that are in both the bag being called and in the bag being submitted. For example, if bag A has:

1 onion
3 carrots
6 peas

and bag B has:

1 carrot
4 peas

Then the Intersection bag would contain:

1 carrots
4 peas

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Bag,In)Bag:P%Bag

Method: RemoveAll

Description: Removes all the Bag items and sets the cardinality to 0. If AutoDestroy is true, all objects pointed to will be destroyed as well.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RemoveAllOccurrences

Description : Removes all items from a collection that allows duplicates. This abstract method is overriden in subclasses that allow duplicates. For those collections that do not allow duplicates, this method calls RemoveElement.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)o:P%Item,
(Required,Type=Variant,In)Item:P%Item

Method: RemoveElement

Description: Removes the specified item from the collection. RemoveElement finds the element by its value and removes it. Note, if there are two identical elements in the array, only the first one is removed.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Item

Method: RemoveElementAt

Description: RemoveElementAt positionally removes an element from the bag independent of its value. An iteration object must be used to specify the position. No presumption of order can be made for the next element to be processed by the iterator. If two or more identical elements exist in the bag, only one is removed.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Base$Bag>Iterator,In)CurrentPosition:P%Iter

Method: ReplaceElementAt

Description: Replaces the element pointed to by the position iterator with the item passed in. An iterator object must be specified.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Variant,In)o:P%Item,
(Optional,Type=Base$Array>Iterator,In)CurrentPosition:P%Iter

Method: RetrieveElementAt

Description: RetrieveElementAt retrieves an element of the array based on it's position.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Bag>Iterator,In)CurrentPosition:P%Iter

Method: Union

Description: Union returns a new bag object which contains bag elements that are in the bag being called or in the bag being submitted or both. For example, if bag A has:
1 onion
3 carrots
6 peas
and bag B has:
1 carrot
4 p
eas
Then the Union bag would contain:
1 onion
4 carrots
10 peas

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Bag,In)Bag:P%Bag

Property: AllowsDuplicates

Description: AllowsDuplicates is a read-only property that indicates whether or not duplicates are allowed in the given collection. Bags allow duplicates.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IsOrdered

Description: IsOrdered returns a bollean value. A 0 is returned If the collection is unordered. A 1 is returned If the collection is ordered. Bags are unordered collections and this property will return a 0.

"Ordered" is defined to reflect the constancy of retrieval. For example, if an iterator is used to retrieve the next element from a specified position in an array, it can be assumed that the same data element will be presented each time the call is made-- provided there are no inserts or removes done between the calls.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Interface: VariableFactory

Method: HasNull

Options: Public,Method,Void
Parameters: None

Instance Variables:

I%HasNull

Description: HasNull stored the count of null items stored by the iterm.

Inheritable: Yes
Initialization: Initialized
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%IC

Description: A builtin array used to count the instances of a given element in the bag.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Bag>Iterator

Description: A BagIterator object contains iterator functions that operate specifically on a Bag collection. It can only be created through the CreateIterator method in the Primary interface of the Bag object.

Type: Concrete
Superclasses: Collection>Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the Bag iterator.

Method: CREATE

Description: General CREATE command for iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the appropriate collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: Last

Description: Last returns the last element of the Bag collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Next

Description: Next returns the next element in the Bag collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

I%HasNull

Description: HasNull indicates whether the collection has null keys.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%ItemNumber

Description: Contains the number of the current item that has been seen.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%RemoveList

Description: Contains a list of items that have been removed during the iteration process.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%SetAsideList

Description: Contains a list of items that are set aside during the iteration process.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%Status

Description: I%Status contains the status of the iterator. It can be "Initial", "Active" or "nil".

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$BagIterator

Description: A BagIterator object contains iterator functions that operate specifically on a Bag collection. It can only be created through the CreateIterator method in the Primary interface of the Bag object.

Type: Concrete
Superclasses: Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the Bag iterator.

Method: CREATE

Description: General CREATE command for iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the appropriate collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called.

Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: Last

Description: Last returns the last element of the Bag collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Next

Description: Next returns the next element in the Bag collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

I%HasNull

Description: HasNull indicates whether the collection has null keys.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%ItemNumber

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%RemoveList

Description: Contains a list of items that have been removed during the iteration process.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%SetAsideList

Description: Contains a list of items that are set aside during the iteration process.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%Status

Description: I%Status contains the status of the iterator. It can be "Initial", "Active" or "nil".

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Collection

Description: An abstract class that defines interfaces and services for specific collection subclasses.

Type: Abstract
Superclasses: AbsLockableObject

Interface: Factory

Description: The Factory interface contains all constructor and destructor services for the Collection subclasses.

Method: CREATE

Description: CREATE creates the basic collection. P%Data is a list of element to be inserted. P%Data itself contains the number of entries in the P%Data array. The different subclasses of collection objects handles any order or key storage necessary to the object. Note that data elements may or may not be passed in. The collection is created regardless.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: DESTROY

Description: DESTROY destroys the collection. All iterators are removed.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: DestroyAll

Description: The DestroyAll method is an internal method that destroys all elements of the collection. It is used to clear the collection of elements for reuse.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: The Internal interface contains all private services for the Collection subclasses.

Method: CopyExternally

Description: Provides a common copy function that uses the public interface to transfer the contents of one Collection object to another of a different Collection type. Each Copy method of the subclasses call this method if the target object passed into that method is not the same type as the source object.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Collection,In)Target:P%Target

Method: GetPointer

Description: Returns the object (M) pointer of the current object.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: RemoveIterator

Description: Removes a specified iterator entry in the iterator list when called by the iterator itself. Assumes that the caller is an iterator of this collection.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Interface: Primary

Description: The Primary interface contains all public services for the Collection subclasses.

Event: AllRemoved

Description: Event thrown when all elements in the collection are removed. The event passes the collection OID and the event name to the watcher.

Callback Documentation:

```
AllRemov(OID,Event)          ; all elements removed from a collection
        ; OID - the collection object which was cleaned
        ; Event - the name of the event "AllRemoved"
```

Event: ElementAdded

Description: The ElementAdded event is thrown when an element is added to the collection. The event passes the collection IOD, the event name and the element added to the watcher.

Callback Documentation:

```
EleAdded(OID,Event,Item)    ; element added to a collection
        ; OID - the collection object in which an element was added.
        ; Event - the name of the event "ElementAdded".
        ; Item - the item added to the collection.
```

Event: ElementRemoved

Description: The ElementRemoved event is thrown when an element is removed from the collection. The event passes the collection OID, the event name and the element that has been removed to the watcher.

Callback Documentation:

```
EleRem(OID,Event,Item)      ; element removed from a collection
        ; OID - the collection object in which an element was removed.
        ; Event - the name of the event "ElementRemoved".
        ; Item - the item removed.
```

Method: ContainsElement

Description: Determines whether the collection contains a specific element. Each subclass implements this method accordiing to its specific structure.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=String,In)o:P%Item

Method: Copy

Description: Copies the elements of a collection into a second (target) collection.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Object,In)P%CopyTarget

Method: CreateIterator

Description: Creates an iterator object that is used to traverse a collection. Used as an abstract template for collection subclasses.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: InsertElement

Description: InsertElement inserts a new element into the collection. This method serves as an abstract template for all subclasses. The means by which the element is inserted is determined by the collection subclass.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Object,In)o:P%Object

Method: InsertElementAt

Description: InsertElementAt inserts the data element in the collection at a specified position. This method is abstract method that serves as a template for Collection subclasses.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=Variant,In)Value:P%value,
(Required,Type=Variant,In)Key:P%pos

Method: KeyType

Description: Determines the type of collection:
0 - No key - (Set, List, etc.).
1 - Keyed and manually maintained (Map, MultiMap).
2 - Keyed and automatically maintained (Dictionary).

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: RemoveAll

Description: Removes all the collections items and sets the cardinality to 0. If AutoDestroy is true, all objects pointed to will be destroyed as well.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: RemoveAllOccurrences

Description : Removes all items from a collection that allows duplicates. This abstract method is overriden in subclasses that allow duplicates. For those collections that do not allow duplicates, this method calls RemoveElement.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=Variant,In)o:P%Item,
(Required,Type=Variant,In)Item:P%Item

Method: RemoveElement

Description: Removes the specified item from the collection. RemoveElement finds the element by its value and removes it. Note, if there are two identical elements in the array, only the first one is removed.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Variant,In)o:P%Item

Method: RemoveElementAt

Description: RemoveElementAt positionally removes an element. The definition of the position is dependant on the collection subclass. It could refer to a key or an index. The current position of the associated iterator can also be used.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Object,In)CurrentPosition:P%Iterator

Method: ReplaceElementAt

Description: Replaces the element pointed to by the position iterator with the item passed in.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=Variant,In)o:P%Item,
(Required,Type=Object,In)CurrentPosition:P%Iterator

Method: RetrieveElementAt

Description: RetrieveElementAt retrieves an element of the array based on it's position.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)Position:P%Pos

Property: AllowsDuplicates

Description: AllowsDuplicates is a read-only property that indicates whether or not duplicates are allowed in the given collection.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: AutoDelete

Description : A Read/Write boolean property that when TRUE will cause the collection to delete itself when the last iterator has been removed.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All,Abstract
Parameters: (Required,System,Type=String,In)P%Value

Property: AutoDestroy

Description: If AutoDestroy is set to true, the objects in the collection will be automatically destroyed when the collection is destoryed or a RemoveAll is invoked.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All,Abstract
Parameters: (Required,System,Type=Boolean,In)I%AutoDestroy

Property: Capabilities

Description: Returns a number where each power of 2 bit in the number represents a capability that the collection object has.

Bit Mask

0x0001 Ordered (1)
0x0002 Duplicates (2)
0x0004 Insertion (4)
0x0008 Removal (8)
0x0010 Open Extent (16)
0x0040 Indexable (64)
0x0080 Keyed (128)
0x1000 Auto Delete (4096)
0x2000 Auto Destroiy (8192)

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: Cardinality

Description: Cardinality returns the number of elements in the collection. This value is changed as elements are inserted or removed.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: IsEmpty

Description: The abstract IsEmpty property returns a numeric value that idicates whether the collection is empty (0) or contains items (>0).

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: IsKeyed

Description: The abstract IsKeyed property returns a boolean value that idicates whether the collection items have key values.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: IsOrdered

Description: The abstract IsOrdered property returns a bollean value. A 0 is returned If the collection is unordered. A 1 is returned If the collection is ordered.

"Ordered" is defined to reflect the constancy of retrieval. For example, if an iterator is used to retrieve the next element from a specified position in an array, it can be assumed that the same data element will be presented each time the call is made-- provided there are no inserts or removes done between the calls.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: Type

Description: Returns the value of I%Type that contains the type of collection.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,System,Type=Variant,In)Type:I%Type

Instance Variables:

I%AutoDelete

Description: A boolean value (defaults to zero), tied to the AutoDelete property. When this is TRUE, the collection will delete itself when the last iterator is removed.

Inheritable: Yes
Initialization: Static
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%AutoDestroy

Description: If set to true the objects in the collection will be automatically destroyed when the collection is destoryed or a RemoveAll is invoked.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%Cardinality

Description: Contains the number of objects stored in the collection.

Inheritable: Yes
Initialization: Static
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%IL

Description: A instance array holding the items inserted in the collection.

Inheritable: Yes

Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%IteratorList

Description: An instance array that keeps track of all iterators currently accessing the collection.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%Type

Description: Stores type of collection.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=No
Creation Keyword: CHILD=1
Creation Parameters:

Class: Base$Collection>Iterator

Description: Iterators are objects that can be used to traverse and retrieve information for collection objects. Iterators have the general methods and properties of order, first, last, next, etc. Each subclass of the collections class has its own type of iterator. For Array collections, there is an Array iterator, for Set collections there is a Set iterator. Because of this it is important for iterators to NEVER be created on their own. The CreateIterator method of the respective collection is always used to create the appropriate iterator. This insures both that the proper type of iterator is created and the object associated with the iterator is the proper object.

Type: Abstract

Interface: Factory

Description: Contains all abstract constructor and destructor services for the iterators.

Method: CREATE

Description: General CREATE command for iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the appropriate collection object.

Options: Public,Method,Void,Platform=All,Abstract

Parameters: (Required,Type=String,In)List:I%List

Method: DESTROY

Description: Destroys the particular iterator associated with the given collection object.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag

object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Property: CurrentPosition

Description: CurrentPosition returns the internal number representing the current iterator's position in the collection object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: Last

Description: Last returns the last element of the collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All,Abstract

Parameters: (Optional,Type=Numeric,In)Position:P%Number

Method: Next

Description: Next returns the next element in the collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: Reset

Description: Reset brings the iterator back to the initial state. Iteration order is maintained.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,System,Type=Variant,In)Order:P%Order

Instance Variables:

I%AddList

Description: List of collection items that have been added to the collection while iterating it.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%CurrentCell

Description: I%CurrentCell contains a pointer to the current cell of the currently pointed to item.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%CurrentItem

Description: I%CurrentItem contains a pointer to the current item.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%IOrder

Description: I%IsOrder specifies the iteration order: Forward or Backward. When the iterator is created, the variable defaults to "Forward". Property "IterationOrder" can change this value.

Inheritable: Yes
Initialization: Static
Binding: Expression="Forward"
Creation Options: Manually Maintained=No, Dependent=Yes

I%ItemNumber

Description: I%ItemNumber contains the number of the item currently pointed to.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%List

Description: I%List contains the $POINTER (partial global reference used in subscript indirection) of the collection object.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%OrderKy

Description: I%OrderKy is the $Order key and is linked to IOrder.

If IOrder == "Forward" the Value is 1
If IOrder == "Backward" the Value is -1

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1
Creation Options: Manually Maintained=No, Dependent=Yes

I%Owner

Description: I%Owner is created at interator creation. It contains a pointer back to the owner collection that created the iterator.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=No
Creation Keyword: =
Creation Parameters:

I%RemoveList

Description: List of collection items that have been removed while iterating.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%SetAsideList

Description: List of collection items that have been iterated through and are taken out of the iteration path to avoid redundantly iterating through them again.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%Status

Description: I%Status contains the status of the iterator. It can be "Initial", "Active" or "nil".

Inheritable: Yes
Initialization: Static

Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%StopNumber

Description: I%StopNumber is used internally.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$CollectionProtector

Description: The CollectionProtector is used to wrap collections so as to hide the collections interface and only expose those operations that are safe.

Type: Concrete
Superclasses: Collection

Interface: Factory

Method: CREATE

Description: CREATE creates the collection protector. It takes a single argument which is the collection to wrap & protect.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Variant,In)Collection:P%Col

Method: DESTROY

Description: The DestroyAll method is an internal method that destroys all elements of the collection. It is used to clear the collection of elements for reuse. The protected destructor prevents the the related collection from being effected.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Method: ContainsElement

Description: Determines whether the collection contains a specific element.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=String,In)o:P%Item

Method: ContainsKey

Description: ContainsKey cheks a specified object in the collection for a given key.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%KeyValue

Method: Copy

Description: Copies the elements of a collection into a second (target) collection.

Options: Public,Method,Void,Platform=All,Virtual
Parameters: (Required,Type=Object,In)c2:P%Collection

Method: CreateIterator

Description: Creates a CollectionProtector iterator object that is used to traverse the wrapped collection. Execution of this method is the only way an iterator can be created. The creation is defered to the wrapped collection CreateIterator method.

Options: Public,Method,Void
Parameters: None

Method: InsertElement

Description: When used directly with a collection object, InsertElement inserts a new element into the collection. When the Collection protector is used this operation is an error.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)o:P%Object

Method: Lock

Description: Locks a wrapped Object.

Options: Private,Method,Void,Inheritable,Platform=All
Parameters:
(Optional,Type=String,In)Timeout:P%Timeout,
(Optional,Type=String,In)Interactive:P%Interactive=0

Method: RemoveAll

Description: This is a a protected operation and will assert an error if used.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RemoveElement

Description: This method is not enabled and will generate an error if it is invoked.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Item

Method: RemoveElementAt

Description: Invoking this method will generate an error message.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Base$Bag>Iterator,In)CurrentPosition:P%Iter

Method: ReplaceElementAt

Description: Using this method will produce an error message.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Variant,In)o:P%Item,
(Optional,Type=Base$Collection>Iterator,In)CurrentPosition:P%Iterator

Method: RetrieveElementAt

Description: RetrieveElementAt retrieves an element of the array based on it's position.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Bag>Iterator,In)CurrentPosition:P%Iter

Method: RetrieveElementsByKey

Description: RetrieveElementsByKey returns a list object of all of the objects that satisfy a given key.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%KeyValue

Method: RetrieveElementsByPattern

Description: RetrieveElementsByPattern returns a list object of all objects that satisfy a given M pattern match.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Pattern:P%Pattern

Method: RetrieveKey

Description: RetrieveKey returns the key value by which the object is associated with the dictionary.

Options: Public,Method,Void,Platform=All

Parameters: (Required,Type=Object,In)Object:P%Object

Method: Unlock

Description: Unlocks the wrapped object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)All:P%All=0

Property: AllowsDuplicates

Description: AllowsDuplicates is a read-only property that indicates whether or not duplicates are allowed in the given collection. The value produced will depend on the wrapped collection.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Virtual
Parameters: None

Property: AutoDelete

Description : A boolean property that when TRUE will cause the collection to delete itself when the last iterator has been removed.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Virtual
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All,Virtual
Parameters: (Required,System,Type=String,In)P%Value

Property: Capabilities

Description: Returns a number where each power of 2 bit in the number represents a capability that the collection object has.

Bit Mask

0x0001 Ordered (1)
0x0002 Duplicates (2)
0x0004 Insertion (4)
0x0008 Removal (8)
0x0010 Open Extent (16)
0x0020 Stable (32)

0x0040 Indexable (64)
0x0080 Keyed (128)
0x1000 Auto Delete (4096)
0x2000 Auto Destroiy (8192)

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Virtual,Abstract
Parameters: None

Property: Cardinality

Description: Cardinality returns the number of elements in the collection. This value is changed as elements are inserted or removed.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Virtual
Parameters: None

Property: IsEmpty

Description: The IsEmpty property returns a numeric value from the wrapped collection that idicates whether the collection is empty (0) or contains items (>0).

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IsOrdered

Description: IsOrdered returns a bollean value. A 0 is returned If the collection is unordered. A 1 is returned If the collection is ordered. This property will return the value of the wrapped collection.

"Ordered" is defined to reflect the constancy of retrieval. For example, if an iterator is used to retrieve the next element from a specified position in an array, it can be assumed that the same data element will be presented each time the call is made-- provided there are no inserts or removes done between the calls.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Key

Description: Key returns the name of the property being used as a key for the instance of the variable.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

Class: Base$CompoundCriteria

Description: A compound Criteria, containing multiple component Criteria, and returning true or false based on the truth or falsity of its component criteria.

Type: Abstract
Superclasses: FilterCriteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: DESTROY

Description: Abstract destructor for CompoundCriteria classes. DESTROYing a CompoundCriteria explicitly destroys all the individual criteria of which it is composed.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Method: AddCriteria

Description: Abstract method that dds one or more new criteria to the criteria list.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: RemoveCriteria

Description: Removes one or more criteria from this compound criteria.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: Properties

Description: Returns a collection containing the properties affected by the criterion. An object must implement these properties in order to be used by the Matches method for this class. Since a CompoundCriteria may contain any number of individual Criteria, many different properties may be referenced by it.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Instance Variables:

I%CriteriaList

Description: Contains the list of criteria to be operated on by the Matches method.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$ContainsCriteria

Description: Implements the contains criteria operations.

Type: Concrete
Superclasses: RelationalCriteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Method: Matches

Description: Returns true if a certain property of the specified object contains a certain value. The certain property is defined by the ContainsCriteria's Property property, and the certain value is defined by the ContainsCriteria's Value property.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)o:P%Object

Instance Variables:

Class: Base$ContentHandler

Description: Abstract class for defining content handling of document parsers. The abstract content handler for the document is subclassed to ContentHander. It acts as the specific template for the document type.

Type: Abstract

Class: Base$Criteria

Description: A true/false criterion, determining whether or not some object matches a certain requirement (or possibly a set of requirements.) This abstract class and its descendants were designed for winnowing, to be repeatedly applied to each object in a group (such as a database), in order to evaluate which of them meet a given set of requirements.

Type: Abstract
Superclasses: AbsFactoryObject

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: DESTROY

Description: Abstract destructor for Criteria classes.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the abstract Criteria class.

Method: Matches

Description: Abstract method that performs a matching operation. Returns true if the specified object matches the criteria.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Object,In)o:P%Object

Property: IsRange

Description: Returns true if the criteria are based on a range (exposing RangeStart and RangeEnd properties), false if not. Returns false in this case.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: Properties

Description: Returns a collection containing the properties affected by the criterion. An object must implement these properties in order to be used by the Matches method for this class.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Class: Base$DataManager

Description: The concrete base class DataManager is an aggregate object containing unique, homogeneous objects arranged according to zero or more key property values.

Related Classes: Dictionary, a component class used to organize key property values. Set, since the DataManager's elements are unique.

Usage: DataManager is used for complex organization of data in ways that would not be possible with a simple Dictionary, or other collection. Both the capabilities and overhead of DataManager are greater than those of collections.

Type: Concrete
Superclasses: AbsFactoryObject,AbsLockableObject

Interface: Factory

Description: Interface that holds all construtors and destructors.

Method: DESTROY

Description: If instance control is turned on, then DESTROY will specifically destroy all the items in the DataManager and flush all the dictionaries. Otherwise the data manager is simply destroyed as a normal object.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Private Interface used by the data manager.

Method: Watch

Description: Watch places a watch on either the specified object in the Data Manager or all of the objects in the Data Manager depending on whether or not an object is specified.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Object,In)Object:P%Object

Interface: Primary

Description: Public interface for the DataManager object.

Method: AddKey

Description: Adds a new key property to the data manager. If the specified property is not already being tracked, it will create a new dictionary for that property and copy the items into that new dictionary.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Key:P%KeyName

Method: ContainsElement

Description: Determines whether the collection contains a specific element by delegating the call to the collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Item:P%Item

Method: CreateElement

Description: Creates a new element of the data manager's base class (defined by the Class property).

Options: Public,Method,Void,Platform=All
Parameters: None

Method: CreateIteratorForKey

Description: CreateIteratorForKey returns an instance of an iterator pointing to the dictionary for the specified key.
If no keyname is specified, the iterator returned is for the base list of the Data Manager. The base list is a Set object that contains ALL items that have been inserted into the DataManager.

Example: The following example retrieves an iterator for a dictionary that tracks "LastName" in the DataManager
Set I%iter=I%DataM.CreateIteratorForKey(KeyName:"LastName")

Options: Public,Method,Void,Platform=All

Parameters: (Optional,Type=String,In)Key:P%KeyName=""

Method: InsertElement

Description: Adds an element to the Data Manager, which also causes it to be added to all the dictionaries in the dictionary list.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)o:P%Item

Method: RemoveElement

Description: Removes the specified item from the data manager (and its associated dictionaries). Destroys the item if instance control is turned on.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)o:P%Item

Method: RemoveKey

Description: Removes a key from the data manager's list of key properties.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Key:P%KeyName

Method: RetrieveElementsByKey

Description: RetrieveElementsByKey returns a list object of all of the objects that satisfy a given key/value within the DataManager.
Example: Below, assume that I%oid points to a DataManager object that tracks items by "LastName". Upon completion, I%Loid will be a list object that contains all objects from the DataManager that have a LastName property equal to "Smith".

Set I%Loid=I%oid.LookupItem(Key:"LastName",Value:"Smith")

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=String,In)Key:P%KeyName,
(Required,Type=String,In)Value:P%Value

Method: SelectMatches

Description: Returns the set of elements matching the specified criteria for the specified ranges.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Collection,In)Criteria:P%CriteriaList

Property: Cardinality

Description: Returns a non-negative integer, the number of elements in the data manager.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Class

Description: Returns the name of the base class: items inserted into the data manager are presumed to be of this class. This class will be used as a default when new items are created, and any items inserted into the DataManager must implement all of its key properties.

Examples:
Creation: CREATE I%DM=Base$DataManager:Share=1:"Patient"
Value: Set T%DMClassName=I%DM.Class

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Create

Options: Public,Property_Create,Void,Platform=All
Parameters: (RequiredRequired,System,Type=String,In)Value:P%Value

Property: ControlsData

Description: True if the items inserted into the data manager will be treated as components. In this case, the data manager will destroy all items when they are removed, or when the data manager itself is destroyed.

Example:
Assign: SET I%DM.ControlsData=1
Value: IF 'I%DM.ControlsData DESTROY A%ThisElement

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All
Parameters: (Required,System,Type=Boolean,In)Value:P%Value

Property: InitialKey

Description: Creates the initial Key.

Options: Public,Inheritable

Accessor: Create

Options: Public,Property_Create,Void,Platform=All
Parameters: (RequiredRequired,System,Type=String,In)Value:P%Value

Property: Keys

Description: An array of keys suitable for $Ordering, subscripted by key property name.

Options: Public,Inheritable

Accessor: $Order

Options: Public,Property_Order,Void,Platform=All
Parameters:
(RequiredRequired,System,Type=Numeric,In)Direction:P%Dir,
(Required,Type=String,In)Subscript:P%Sub

Instance Variables:

I%BaseClass

Description: Contains the name of the base class: items inserted into the data manager are presumed to be of this class. This class will be used as a default when new items are created, and any items inserted into the DataManager must implement all of its key properties.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%BaseList

Description: A set containing the actual items inserted into the data manager.

Inheritable: Yes
Initialization: Dynamic
Binding: Base$Set
Creation Options: Manually Maintained=No, Dependent=Yes

I%DictionaryList

Description: Contains the dictionaries used for cross-referencing.
I%DictionaryList(Key)=Dictionary object

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%InstanceControl

Description: Corresponds to the ControlsData property. 1 by default.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Date

Description: Date immutables collect under one roof all of the different operations concerned with dates. These include date validation, date arithmetic and date formatting as part of the immutable date class.

Type: Concrete
Superclasses: Immutable

Interface: Factory

Description: Interface the contains constructors and desctructors.

Method: CREATE

Description: CREATE creates the immutable date object. P%Data is the date to be associated with the object at time of creation. If no date is passed in, the current date is used. Dates can be submitted in a variety of forms: a date object, timestamp obje ct and miscellaneous external formats such as "10/12/60", "T-1", "TODAY", etc. Note: there is no provision to determine if the created object is valid or not. This is determined by the IsValid method.Generally, the possible valid inputs to create a date follow the underlying M date utility formats. For example, on the MSM platform, all inputs to the ^%DI utility are accepted here. In addition you can specify "Today", TOMORROW", or "YESTERDAY" (not case sensitive). For European format: Specify European format as DD-MMM-YY for example "12-OCT-1960". You can specify the full month name, but in either case, the month must be in capital letters. Also, spaces may be used instead of dashes to separate the fields.

Example:CREATE I%oid=Base$Date("10/2/1960")

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Date:P%Date=$Horolog

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: This is the public interface for all Date services.

Method: Add

Description: Add performs an aritmetic add to the current date object and returns a new object with the new date.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Numeric,In)Days:P%Days=0,
(Optional,Type=Numeric,In)Interval:P%Inter

Method: DaysInMonth

Description: DaysInMonth returns the number of days in either the month of the object (if no value is passed in), the days in in the month of the date in the date object passed in, or the days in the month of the year and month passed in.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Variant,In)Date:P%Date=%objcf(%objsp,0),
(Optional,Type=String,In)P%Month

Method: DaysInYear

Description: DaysInYear returns the number of days in either the year of the object (if no value is passed in), the days in in the year of the date in the date object passed in, or the days in the year of the digit year passed in.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Date:P%Date=%objcf(%objsp,0)

Method: Decrement

Description: Decrement returns a new object whose date value is exactly one day less than the date value of the current object called or the date value is decremented by the value passed in.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Numeric,In)Decrement:P%Decr=1

Method: FromString

Description: Constructs a Date Object from a String. Returns a Null sting if the String is invalid.

1)julian date (future)
2)misc external formats (eg "T-1", "TODAY", "TOMORROW, "YESTERDAY", "10/12/60", etc)
3)other - defaults to TODAY
4) $H Days.
5) OQL Format Date YYYY-MM-DD

Options: Public,Method,Void,Static,Platform=All
Parameters: (Optional,Type=String,In)Date:P%Date=+$Horolog

Method: GreaterThan

Description: GreaterThan compares the input date specification to the date value contained within the date object. The method returns a truth value whether or not the date object is greater than the input specification.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Date:P%Date

Method: GreaterThanOrEqual

Description: GreaterThanOrEqual compares the input date specification to the date value contained within the date object. The method returns a truth value whether or not the date object is greater than or equal the input specification.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Date:P%Date

Method: Increment

Description: Increment returns a new object whose date value is exactly one day more than the date value of the current object called or the date value is incremented by the value passed in.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Numeric,In)Increment:P%Incr=1

Method: IsBetween

Description: IsBetween determines if the date value contained in the date object is between the two dates described in the two objects passed in to the method.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Base$Date,In)StartDate:P%StartDate,
(Required,Type=Base$Date,In)EndDate:P%EndDate

Method: IsEqual

Description: IsEqual performs a logical date compare operations between the current date object and the submitted date description. The submitted date description can be a date object or a text date description such as "YESTERDAY", "T+1". etc.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Date:P%Date

Method: IsValid

Description: IsValid allows the user to determine if the contents of the date object is valid or not. It is possible in the CREATE to create a date object with an invalid value.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: IsValidDate

Description: IsValidDate determines whether or not the input date is in fact a valid date or not.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Variant,In)Date:P%Date=%objcf(%objsp,0),
(Optional,Type=Numeric,In)Month:P%Mnth,
(Optional,Type=Numeric,In)Day:P%Dia

Method: LessThan

Description: LessThan compares the input date specification to the date value contained within the date object. The method returns a truth value whether or not the date object is less than the input specification.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Date:P%Date

Method: LessThanOrEqual

Description: LessThanOrEqual compares the input date specification to the date value contained within the date object. The method returns a truth value whether or not the date object is less than or equal the input specification.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Date:P%Date

Method: Next

Description: Next returns a date object whose date value is the next date forward in time that corresponds to the nearest day of the week passed in. For example, if the contained date is Wednesday (03) and the desired future day is Monday (01), then the new date object will have a date corresponding to the following Monday.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)Weekday:P%Weekday

Method: NotEqual

Description: NotEqual compares the input date specification to the date value contained within the date object. The method returns a truth value whether or not the date object is not equal to the input specification.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Date:P%Date

Method: Overlaps

Description: Overlaps takes any combination of two date objects or two time stamp objects and compares them to determine if the two ranges overlap.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Object,In)Parm1:P%Parm1="",
(Optional,Type=Object,In)Parm2:P%Parm2="",
(Optional,Type=Object,In)Parm3:P%Parm3="",
(Optional,Type=Object,In)Parm4:P%Parm4=""

Method: Previous

Description: Previous returns a date object whose date value is the previous date backward in time that corresponds to the nearest day of the week passed in. For example, if the contained date is Wednesday (03) and the desired past day is Monday (01), then the new date object will have a date corresponding to the previous Monday.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)Weekday:P%Weekday

Method: Subtract

Description: Subtract performs an aritmetic subtraction to the current date object and returns a new object with the new date.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Numeric,In)Days:P%Days=0,
(Optional,Type=Numeric,In)Interval:P%Inter

Property: Current

Description: Current returns the text value of the date of this object. (This is Y2K compatible.)

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Day

Description: Day returns the numeric day of the month of the date contained in the date object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: DayOfWeek

Description: DayOfWeek returns the text day of the week of the date that is contained in the date object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: DayOfYear

Description: DayOfYear returns the number of days from the beginning of the year for the date contained in the date object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IsLeapYear

Description: IsLeapYear returns whether or not the date contained in the date object is in a leap year or not. Value property: Truth: 1 if leapyear, 0 if not.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Mdate

Description: Mdate returns the $H format date for operational purposes.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Month

Description: Month returns the numeric month of the date contained in the date object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: TextDate

Description: TextDate returns a text date based on the value contained in the date object. If the date is not valid, a null is returned.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: TextMonth

Description: TextMonth returns the text month (JAN, FEB, etc) for the month of the date object. If the date in the date object is not valid, null is returned.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Today

Description: Today is the property that returns the text value of the current date (today) regardless of the date that is in the date object. This is provided as a convenience to the user. (This is Y2K compatible.)

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Year

Description: Year returns the numeric year of the date contained in the date object as YYYY.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Class: Base$Descriptors

Description: Abstract class for Descriptior types.

Type: Abstract

Class: Base$Dictionary

Description: A Dictionary collection of objects is retrievable by a "key". The key is not part of the Dictionary. Instead, it is a property of the objects collected under the auspices of the Dictionary. Consequently, the Dictionary can be defined as a collection of homogenous objects in that they all have a commonly defined property. For example, a dictionary can be defined to be a collection of objects having the property "Name" and could contain the objects:

Person
Ship
Dog
Cat

Spacecraft
Town
Country

All of which have the common property "Name". (E.G., Person.Name,Cat.Name,etc.)

Type: Concrete
Superclasses: KeyedCollection

Interface: Factory

Description: Contains all constructor and destructor services for the Dictionary iterator.

Method: CREATE

Description: CREATE creates the dictionary object. P%PropKey contains the name of the property to be used for the dictionary key. It is required. P%Data is an optional list of element to be inserted. P%Data itself contains the number of entries in the P%Data array. Note that data elements may or may not be passed in. The dictionary is created regardless.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%PropKey

Method: DESTROY

Description: DESTROY destroys the dictionary and its contents. First, all watches are removed. Then the items are deleted. After this, the iterators are removed.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DestroyAll

Description: The DestroyAll method is an internal method that destroys all elements of the collection. It is used to clear the collection of elements for reuse.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Property: Key

Description:
Key returns the name of the property being used as a key for the instance of the variable.

Access: value
Value format: name of the property
Example:
S property=I%oid.FactoryKey
Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void
Parameters: None

Interface: Internal

Method: KeyModified

Description: KeyModified is called from a watch on the component of the object being used as a key. When the key is modified, this method is called to handled the key structure maintained by the dictionary object so that other operations are done appropriately.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Watch

Description: Watch places a watch on either the specified object in the dictionary or all of the objects in the dictionary depending on whether or not an object is specified. This causes the KeyModified event to occur if the key property in the object is modified.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Object,In)Object:P%Object

Interface: Primary

Method: ContainsElement

Description: Determines whether the collection contains a specific element.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)P%Object

Method: ContainsKey

Description: ContainsKey returns whether or not a key exists in the Dictionary for any item. This is useful to detect if a particular key is in use in the Dictionary.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%KeyValue

Method: Copy

Description: Copies the elements of an Dictionary collection into a target collection. The target collection may be any collection type compatible with the Dictionary collection.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)Dictionary:P%Target

Method: CreateIterator

Description: Creates a Dictionary iterator object that is used to traverse an Dictionary collection. Execution of this method is the only way an iterator can be created.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InsertElement

Description: InsertElement inserts a new element into the Dictionary collection.
Note:
1) null keys can be used.
2) an object can only be entered in the Dictionary once. Multiple additions of the same object are not tracked. If it's there, the insert is not executed and no error condition is returned or created.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Object:P%Object

Method: KeyType

Description: Determines the type of collection:
0 - No key - (Set, List, etc.).
1 - Keyed and manually maintained (Map, MultiMap).
2 - Keyed and automatically maintained (Dictionary).

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RemoveAll

Description: Removes all the Dictionary items and sets the cardinality and to 0. If AutoDestroy is true, all objects pointed to will be destroyed as well.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RemoveElement

Description: Removes the specified item from the collection. RemoveElement finds the element by its value and removes it. Note, if there are two identical elements in the array, only the first one is removed.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Item

Method: RemoveElementAt

Description: RemoveElementAt positionally removes an object from the dictionary. An iteration object must be used to specify the position.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Dictionary>Iterator,In)CurrentPosition:P%Iter

Method: RetrieveElementAt

Description: RetrieveElementAt retrieves an element of the array based on it's position.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Dictionary>Iterator,In)CurrentPosition:P%Iter

Method: RetrieveElementsByKey

Description: RetrieveElementsByKey returns a list object of all of the items that satisfy a given key. The first position of the list contains the key passed in.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%KeyValue

Method: RetrieveElementsByPattern

Description: RetrieveElementsByPattern returns a list collection filled items whose key satisfy a given M pattern match.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Pattern:P%Pattern

Method: RetrieveKey

Description: RetrieveKey returns the key value by which the object is associated with the dictionary.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)Object:P%Object

Property: AllowsDuplicates

Description: AllowsDuplicates is a read-only property that indicates whether or not duplicates are allowed in the given collection. Dictionary objects allow duplicates.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IsOrdered

Description: IsOrdered returns a bollean value. A 0 is returned If the collection is unordered. A 1 is returned If the collection is ordered. Dictionaries are always order and will return 1.

"Ordered" is defined to reflect the constancy of retrieval. For example, if an iterator is used to retrieve the next element from a specified position in an array, it can be assumed that the same data element will be presented each time the call is made-- provided there are no inserts or removes done between the calls.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Key

Description: Key returns the name of the property being used as a key for the instance of the variable.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Value:P%Value

Accessor: Create

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Value:P%Value

Instance Variables:

I%Key

Description: Key contains the name of the property by which inserted objects are ordered. For example, "ZipCode" would mean that objects inserted into this collection are

ordered by their "ZipCode"property. Inserted objects MUST have a property of this key in their Primary interface otherwise the InsertElement method will not insert the object.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%NIL

Description: NIL is used to store objects whose Key property is NULL.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%XIL

Description: XIL is used to store objects in order of the Key property. If the objects' Key property is NULL, the object is stored in the NIL array.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Dictionary>Iterator

Description: A DictionaryIterator object contains iterator functions that operate specifically on a Dictionary collection. It can only be created through the CreateIterator method in the Primary interface of the Dictionary object.

Type: Concrete
Superclasses: KeyedCollection>Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the Dictionary iterator.

Method: CREATE

Description: CREATE command for dictionary iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the dictionary collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Variant,In)Key:P%Key

Property: CurrentPosition

Description: CurrentPosition returns the internal number representing the current iterator's position in the collection object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Interface: Primary

Method: First

Description: First returns the first element of the given Dictionary collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the Dictionary collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the Dictionary collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Next

Description: Next returns the next element in the Dictionary collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Reset

Description: Reset brings the iterator back to the initial state. Iteration order is maintained.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: CurrentKey

Description: The CurrentKey property assigns and retrieves the value of the key being used by the iterator.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: EndKey

Description: The EndKey defines the end of the traverse, which can be at either end of the dictionary depending on the traverse order. NOTE: Changing the EndKey value does NOT change the current key position of the iterator. This is true even if changing the EndKey value puts the position beyond the end of the range.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Order:P%Order

Property: StartKey

Description: The StartKey defines the beginning of the key traversal, which can be at either end of the dictionary depending on the traverse order.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Instance Variables:

I%CurrentKey

Description: The last key that the iterator pointed to. The position need not exist.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%EndKey

Description: The ending key (which may or may not exist) that iteration will end on.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%StartKey

Description: The starting key (which may or may not exist) that iteration will begin on.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$DictionaryIterator

Description: A DictionaryIterator object contains iterator functions that operate specifically on a Dictionary collection. It can only be created through the CreateIterator method in the Primary interface of the Dictionary object.

Type: Concrete
Superclasses: KeyedIterator

Interface: Factory

Description: Contains all constructor and destructor services for the Dictionary iterator.

Method: CREATE

Description: CREATE command for dictionary iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the dictionary collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Variant,In)Key:P%Key

Property: CurrentPosition

Description: CurrentPosition returns the internal number representing the current iterator's position in the collection object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Interface: Primary

Method: First

Description: First returns the first element of the given Dictionary collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the Dictionary collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the Dictionary collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Next

Description: Next returns the next element in the Dictionary collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Reset

Description: Reset brings the iterator back to the initial state. Iteration order is maintained.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: CurrentKey

Description: The CurrentKey property assigns and retrieves the value of the key being used by the iterator.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: EndKey

Description: The EndKey defines the end of the traverse, which can be at either end of the dictionary depending on the traverse order. NOTE: Changing the EndKey value does NOT change the current key position of the iterator. This is true even if changing the EndKey value puts the position beyond the end of the range.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Order:P%Order

Property: StartKey

Description: The StartKey defines the beginning of the key traversal, which can be at either end of the dictionary depending on the traverse order.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Instance Variables:

I%CurrentKey

Description: The last key that the iterator pointed to. The position need not exist.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%EndKey

Description: The ending key (which may or may not exist) that iteration will end on.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%StartKey

Description: The starting key (which may or may not exist) that iteration will begin on.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$DocBuilder

Description: Primary class used to generate an RTF documentation file for a specified EsiObjects heirarchy.

Type: Concrete
Superclasses: Tools

Interface: Factory

Description: Interface the contains constructors and desctructors for the DocBuilder class.

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The Primary interface holds all public services for the DocBuilder class.

Method: GenerateDoc

Description: The primary method that generates all documentation for specified EsiObjects stucture.

How to Use: To use DocBuilder to generate an RTF document for EsiObjects libraries or classes, follow these steps:

1) Bring up the Xecute Shell.

2) Enter the following commands:

Create I%Doc=Base$DocBuilder
Do I%Doc.GenerateDoc($Lib("MyLib"),"C:\MyDir\Library.rtf")

where *$Lib("MyLib")* identifies your library of choice.
If you want to document a single class, use *$Lib("MyLib").FindClass("MyClass")*

"C:\MyDir\Library.rtf" identifies the directory path to the file.

Options: Public,Method,Void,Inheritable,Platform=All
Parameters:
(Required,Type=String,In)ApexObject:P%Apex,
(Optional,Type=String,In)File:P%File

Instance Variables:

Class: Base$ExactHitCriteria

Description: A Criterion that returns true if an object's property is exactly equal to a certain value.

Type: Concrete
Superclasses: FilterCriteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: CREATE

Description: The constructor method is used to create any required structures of the object at creation time. This generally includes all static members, and all relationships.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)Property:I%MatchProperty,
(Optional,Type=String,In)Value:I%MatchValue

Method: DESTROY

Description: Destructor for ExactHitCriteria class.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitClassVars

Options: Public,Method,Void

Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Method: Matches

Description: Returns true if a certain property of the specified object is equal to a certain value. The certain property is defined by the ExactHitCriteria's Property property, and the certain value is defined by the ExactHitCriteria's Value property.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)o:P%Object

Property: Properties

Description: Returns a collection containing the properties affected by the criterion. An object must implement these properties in order to be used by the Matches method for this class.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Property

Description: Sets and returns the property name the Matches method will reference to determine whether it equals the value.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All
Parameters: (Required,System,Type=String,In)Property:P%Prop

Property: Value

Description: The exact value that will be compared to an object's property by the Matches method.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All
Parameters: (Required,System,Type=String,In)Value:P%Val

Instance Variables:

I%MatchProperty

Description: Contains the name of the property to be matched. The Match method will reference this property on the specified object.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%MatchValue

Description: Contains the value to which the object's property must be equal, in order to satisfy the ExactHitCriteria.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$FilterCriteria

Description: A general grouping of Criteria classes specifically devoted to filtering objects. Each object is evaluated according to a specific criterion (or compound set of criteria).

Type: Abstract
Superclasses: Criteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Class: Base$GreaterThanCriteria

Description: Implements the greater than criteria operations.

Type: Concrete
Superclasses: RelationalCriteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Method: Matches

Description: Returns true if the specified object property is greater than the specified criteria.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)o:P%Object

Instance Variables:

Class: Base$Immutable

Description: The Immutables class involves objects that, once created, are not changeable until the object is destroyed. These objects are concerned with time including dates, times, ranges, stamps and intervals. There is no commonality in methodology between the different immutable classes. The only common ground between them is their immutability. There are five subclasses of immutables:

Date	Handles date methods and properties
Time	Handles time methods and properties
Interval	Handles differences between dates and times
TimeRange	Handles coupled time/date ranges with beginning and end values
TimeStamp	Handles a given time/date combination

The methods of immutables by and large process and return objects. For exampe. using an interval method to determine the difference between two date objects returns a new object describing the interval.

Type: Abstract

Interface: Factory

Description: Interface the contains constructors and desctructors.

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Class: Base$InternalStream

Description: Provides a generic IO interface to internal, local data based on the M Open, Use, Read, Write and Close commands.

Type: Concrete
Superclasses: Stream

Interface: Factory

Description: Interface the contains constructors and desctructors.

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: This is the public interface for all abstract InternalStream services.

Method: Read

Description: Reads the specified number of characters from the stream.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)Len:P%Len

Method: ReadLine

Description: Reads a ine from the Stream (Delimited by cr/lf)

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Reset

Description: Reads the specified number of characters from the stream.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Write

Description: Writes data to the Stream.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Text:P%Text

Method: WriteLine

Description: Writes to a Stream apennding a cr/lf.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Line:P%Line

Instance Variables:

I%Blocks

Description: Internal count of the number of $Maxstr blocks are present.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%CurBlock

Description: Identifies the "current" block of ESI$Text from which the stream reader will access data.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1
Creation Options: Manually Maintained=No, Dependent=Yes

I%CurChar

Description: Identifies the "current" character within the current block of ESI$Text from which the stream reader will access data.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1
Creation Options: Manually Maintained=No, Dependent=Yes

I%D

Description: The Data Store for the stream.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%LastLen

Description: The size of the final text block.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%Length

Description: The total number of characters in the text.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Interval

Description: Intervals are descriptions of the time between events. Intervals function similarly to integers in that they can have arithmetic operations, be greater or lesser than one another, etc. Intervals are created from time values, either other intervals, time stamps, or absolute combinations of days, hours, minutes and seconds. Intervals can be a means of detecting differences between time entities in the form of time stamps.

Type: Concrete
Superclasses: Immutable

Interface: Factory

Description: The interface that contains all constructors and destructors.

Method: CREATE

Decscription: CREATE is used to create intervals. A number of inputs can be used in the interval creation. These are checked and processed in the following order:
interval object starting timestamp, ending timestamp
explicit # of days, hours, minutes, seconds.

Any one of the above sets of inputs are interpreted into an interval.

Example:
CREATE I%oid=Base$Interval(Interval:I%interval)
CREATE I%oid=Base$Interval(StartTime:I%ts1,EndTimeI%ts2)
CREATE I%oid=Base$Interval(5,12,30,30)
Create I%oid=Base$Interval(Day:15,Hours:12,Minute:30,Second:30)

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Integer,In)P%Day=0,
(Optional,Type=Integer,In)P%Hour=0,
(Optional,Type=Integer,In)P%Minute=0,
(Optional,Type=Integer,In)P%Sec=0,
(Optional,Type=Integer,In)Interval:P%Int,
(Optional,Type=Integer,In)StartTime:P%Start,
(Optional,Type=Integer,In)EndTime:P%End,

(Optional,Type=Integer,In)Second:P%Sec,
(Optional,Type=Integer,In)Minute:P%Minute,
(Optional,Type=Integer,In)Hour:P%Hour,
(Optional,Type=Integer,In)Day:P%Day

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The Public interface for the Interval class.

Method: Add

Description: Add performs an aritmetic add of an interval object to the current interval object and returns a new object with the new interval.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Interval:P%Int

Method: Divide

Description: Divide provides a mechanism to divide the current interval object by either an integer or by the value of a second interval object passed in.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Interval:P%Int

Method: FromString

Description: Construct a Date Object from a String. Returns a Null sting if the String is invalid.

1)julian date (future)
2)misc external formats (eg "T-1", "TODAY", "TOMORROW, "YESTERDAY", "10/12/60", etc)
3)other - defaults to TODAY
4) $H Days.
5) OQL Format Date YYYY-MM-DD

Options: Public,Method,Void,Static,Platform=All
Parameters: (Optional,Type=String,In)Interval:P%Interval=0

Method: GreaterThan

Description: GreaterThan returns a truth value that reflects whether or not the interval object that is passed in is greater than the interval associated with the current interval object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Interval,In)Interval:P%Interval

Method: GreaterThanOrEqual

Description: GreaterThanOrEqual returns a truth value that reflects whether or not the interval object that is passed in is greater than or equal to the interval associated with the current interval object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Interval,In)Interval:P%Interval

Method: IsEqual

Description: IsEqual returns a truth value that reflects whether or not the interval object that is passed in is equal to the interval associated with the current interval object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Interval:P%Interval

Method: LessThan

Description: LessThan returns a truth value that reflects whether or not the interval object that is passed in is less than the interval associated with the current interval object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Interval,In)Interval:P%Interval

Method: LessThanOrEqual

Description: LessThanOrEqual returns a truth value that reflects whether or not the interval object that is passed in is less than or equal to the interval associated with the current interval object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Interval,In)Interval:P%Interval

Method: Mod

Description: Mod provides a mechanism to modulo the current interval object by either an integer or by the value of a second interval object passed in.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Interval:P%Int

Method: Multiply

Description: Multiply provides a mechanism to multiply the current interval object by an integer value passed in.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Interval:P%Int

Method: NotEqual

Description: NotEqual returns a truth value that reflects whether or not the interval object that is passed in is unequal to the interval associated with the current interval object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Interval,In)Interval:P%Interval

Method: Subtract

Description: Subtract provides a mechanism to subtract from the current interval object the value of a second interval object passed in.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Interval:P%Int

Property: Abs

Description: Abs returns an interval object whose value is the absolute value of the current interval object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Day

Description: Day returns the number of days contained in the interval. For example, if an interval contained 3 days, 7 hours, 6 minutes and 3 seconds, Day would return the value 3.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Hour

Description: Hour returns the number of "free" hours contained in the interval. A "free" hour is a hour that is not subsumed into days. For example, if an interval contained 3 days, 7 hours, 6 minutes and 3 seconds, Hour would return the value 7.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IsZero

Description: IsZero returns the truth value reflecting whether or not the contents of the current iterator object is equal to 0.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Minute

Description: Minute returns the number of "free" minutes contained in the interval. A "free" minute is a minute that is not subsumed into the other units of time such as hours or days. For example, if an interval contained 3 days, 7 hours, 6 minutes and 3 seconds, Minute would return the value 6.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Second

Description: Second returns the number of "free" seconds contained in the interval. A "free" second is a second that is not subsumed into the other units of time such as minutes, hours or days. For example, if an interval contained 3 days, 7 hours, 6 minutes and 3 seconds, Second would return the value 3.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: TotalHours

Description: TotalHours returns the total number of hours contained in the interval. For example, if an interval contained 3 days, 7 hours, 6 minutes and 3 seconds, TotalHours would return the value 79.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: TotalMinutes

Description: TotalMinutes returns the total number of minutes contained in the interval. For example, if an interval contained 3 days, 7 minutes, 6 hours and 3 seconds, TotalMinutes would return the value 4746.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: TotalSeconds

Description: TotalSeconds returns the total number of seconds contained in the interval. For example, if an interval contained 3 days, 7 hours, 6 minutes and 3 seconds, TotalSeconds would return the value 284763.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Class: Base$Iterator

Description: Iterators are objects that can be used to traverse and retrieve information for collection objects. Iterators have the general methods and properties of order, first, last, next, etc. Each subclass of the collections class has its own type of iterator. For Array collections, there is an Array iterator, for Set collections there is a Set iterator. Because of this it is important for iterators to NEVER be created on their own. The CreateIterator method of the respective collection is always used to create the appropriate iterator. This insures both that the proper type of iterator is created and the object associated with the iterator is the proper object.

Type: Abstract

Interface: Factory

Description: Contains all abstract constructor and destructor services for the iterators.

Method: CREATE

Description: General CREATE command for iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the appropriate collection object.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=String,In)List:I%List

Method: DESTROY

Description: Destroys the particular iterator associated with the given collection object.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Property: CurrentPosition

Description: CurrentPosition returns the internal number representing the current iterator's position in the collection object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: Last

Description: Last returns the last element of the collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Optional,Type=Numeric,In)Position:P%Number

Method: Next

Description: Next returns the next element in the collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: Reset

Description: Reset brings the iterator back to the initial state. Iteration order is maintained.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,System,Type=Variant,In)Order:P%Order

Instance Variables:

I%AddList

Description: List of collection items that have been added to the collection while iterating it.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%CurrentCell

Description: I%CurrentCell contains a pointer to the current cell of the currently pointed to item.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%CurrentItem

Description: I%CurrentItem contains a pointer to the current item.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%IOrder

Description: I%IsOrder specifies the iteration order: Forward or Backward. When the iterator is created, the variable defaults to "Forward". Property "IterationOrder" can change this value.

Inheritable: Yes
Initialization: Static
Binding: Expression="Forward"
Creation Options: Manually Maintained=No, Dependent=Yes

I%ItemNumber

Description: I%ItemNumber contains the number of the item currently pointed to.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%List

Description: I%List contains the $POINTER (partial global reference used in subscript indirection) of the collection object.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%OrderKy

Description: I%OrderKy is the $Order key and is linked to IOrder.

If IOrder == "Forward" the Value is 1
If IOrder == "Backward" the Value is -1

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1
Creation Options: Manually Maintained=No, Dependent=Yes

I%Owner

Description: I%Owner is created at interator creation. It contains a pointer back to the owner collection that created the iterator.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=No
Creation Keyword: =
Creation Parameters:

I%RemoveList

Description: List of collection items that have been removed while iterating.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%SetAsideList

Description: List of collection items that have been iterated through and are taken out of the iteration path to avoid redundantly iterating through them again.

Inheritable: Yes
Initialization: Static

Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%Status

Description: I%Status contains the status of the iterator. It can be "Initial", "Active" or "nil".

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%StopNumber

Description: I%StopNumber is used internally.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$KeyedCollection

Description: KeyedCollection is an abstract class that contains all collection subclasses that support Name Value entries. That is, each value has one or more keys associated with it.

Type: Abstract
Superclasses: Collection

Interface: Factory

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Method: ContainsKey

Description: ContainsKey returns whether or not a key exists in the KeyedCollection for any item. This is useful to detect if a particular key is in use in the KeyedCollection.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Variant,In)Key:P%KeyValue

Method: KeyType

Description: Determines the type of collection:
0 - No key - (Set, List, etc.).
1 - Keyed and manually maintained (Map, MultiMap).
2 - Keyed and automatically maintained (Dictionary).

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: RetrieveElementsByKey

Description: RetrieveElementsByKey returns a list object of all of the items that satisfy a given key. The first position of the list contains the key passed in. Note: Since Maps cannot have redundant keys, the list will not contain more than one object (plus the key).

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%Key

Method: RetrieveElementsByPattern

Description: RetrieveElementsByPattern returns a list collection filled items whose key satisfy a given M pattern match.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Pattern:P%Pattern

Method: RetrieveKeysByPattern

Description: Returns a list collection filled with keys whose item(s) satisfy a given M pattern match.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Pattern:P%Pattern

Property: IsKeyed

Description: The abstract IsKeyed property returns a boolean value that idicates whether the collection items have key values.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Instance Variables:

Class: Base$KeyedCollection>Iterator

Description: A KeyIterator is an abstract class that contains abstract iterator functions for its subclasses.

Type: Abstract
Superclasses: Collection>Iterator

Interface: Factory

Description: Contains all abstract constructor and destructor services for the keyed iterators.

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Property: CurrentItem

Description: This property retrieves the value of the current item found by the last iterator operation.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Item:P%Item

Property: CurrentKey

Description: The CurrentKey property assigns and retrieves the value of the key being used by the iterator.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All

Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: EndKey

Description: The EndKey defines the end of the traverse, which can be at either end of the dictionary depending on the traverse order. NOTE: Changing the EndKey value does NOT change the current key position of the iterator. This is true even if changing the EndKey value puts the position beyond the end of the range.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: StartKey

Description: The StartKey defines the beginning of the key traversal, which can be at either end of the dictionary depending on the traverse order.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Instance Variables:

Class: Base$KeyedIterator

Description: A KeyIterator is an abstract class that contains abstract iterator functions for its subclasses.

Type: Abstract
Superclasses: Iterator

Interface: Factory

Description: Contains all abstract constructor and destructor services for the keyed iterators.

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Property: CurrentItem

Description: This property retrieves the value of the current item found by the last iterator operation.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Item:P%Item

Property: CurrentKey

Description: The CurrentKey property assigns and retrieves the value of the key being used by the iterator.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: EndKey

Description: The EndKey defines the end of the traverse, which can be at either end of the dictionary depending on the traverse order. NOTE: Changing the EndKey value does

NOT change the current key position of the iterator. This is true even if changing the EndKey value puts the position beyond the end of the range.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: StartKey

Description: The StartKey defines the beginning of the key traversal, which can be at either end of the dictionary depending on the traverse order.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Instance Variables:

Class: Base$LessThanCriteria

Description: Implements the less than criteria operations.

Type: Concrete
Superclasses: RelationalCriteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Method: Matches

Description: Returns true if the specified object property is less than the specified criteria.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)o:P%Object

Instance Variables:

Class: Base$List

Description: A List collection is an ordered collection of objects that can be traversed. This is presented to the outside world as a traditional linked list where each member of the list contain links to the previous and prior members of the collection. Lists are useful for emulating Stacks or Queues since elements can be inserted and retrieved from the head or the tail of the collection.
In addition to having most of the colleciton methods and properties, lists have additional components specific to the nature of a list. This includes being able to insert at the beginning (the "head") or at the end (the "tail") of the list, retrieving at the head or tail, and others.

Type: Concrete
Superclasses: Collection

Interface: Factory

Method: CREATE

Description: CREATE creates the list object. P%Data is an optional list of objects to be inserted. P%Data itself contains the number of entries in the P%Data array. Note that data elements may or may not be passed in. The list is created regardless.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DestroyAll

Description: The DestroyAll method is an internal method that destroys all elements of the collection. It is used to clear the collection of elements for reuse.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Method: ContainsElement

Description: Determines whether the collection contains a specific element.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)o:P%Item,
(Optional,Type=String,In)Item:P%Item

Method: Copy

Description: Copies the elements of an List collection into a target collection. The target collection may be any collection type compatible with the List collection.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)Set:P%Target

Method: CreateIterator

Description: Creates an List iterator object that is used to traverse an List collection. Execution of this method is the only way an iterator can be created.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InsertElement

Description: InsertElement inserts a specified object into the List object at the current position. The current position is volatile. (See property CurrentPosition.)
Note:
1) Insertion in lists only use position. Iterators are not allowed.
2) If the current position is changed while the insertion is being execute, the object may be inserted at an unexpected place. If this occurs, the insertion will NOT be defeated but may have to be searched for.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Item

Method: InsertElementAfter

Description: InsertElement inserts an item into a List after the a specified position. Insertion in lists only uses position. Iterators are not allowed.

Options: Public,Method,Void,Platform=All

Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Required,Type=Numeric,In)Position:P%Pos

Method: InsertElementBefore

Description: InsertElement inserts a specified item into the List before a specified position. Insertion in lists only uses position. Iterators are not allowed.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Required,Type=Numeric,In)Position:P%Pos

Method: InsertFirstElement

Description: InsertFirstElement inserts a specified item passed in as the new head of the list. The old head of the list becomes the next element after the first.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Item:P%Item

Method: InsertLastElement

Description: InsertLastElement inserts a specified item passed in as the new tail of the list. The old tail of the list becomes the next element after the last.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Item:P%Item

Method: RemoveAll

Description: Removes all the List items and sets the cardinality to 0. If AutoDestroy is true, all objects pointed to will be destroyed as well.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RemoveAllOccurrences

Description : Removes all items from a collection that allows duplicates. This abstract method is overriden in subclasses that allow duplicates. For those collections that do not allow duplicates, this method calls RemoveElement.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)o:P%Item,
(Required,Type=Variant,In)Item:P%Item

Method: RemoveElement

Description: Removes the specified item from the collection. RemoveElement finds the element by its value and removes it. Note, if there are two identical elements in the array, only the first one is removed.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Item

Method: RemoveElementAt

Description: RemoveElementAt positionally removes an element from the list. Either an iteration object or an actual position can be specified.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)P%Unknown="",
(Optional,Type=Base$Array>Iterator,In)CurrentPosition:P%Iter,
(Optional,Type=Numeric,In)Position:P%Pos

Method: RemoveFirstElement

Description: RemoveFirstElement removes the first element of the list. The second element of the list becomes the list head.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RemoveLastElement

Description: RemoveLastElement removes the last element of the list. The second to the last element of the list becomes the list tail.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: ReplaceElementAt

Description: Replaces the element pointed to by the position iterator with the item passed in.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Variant,In)Item:P%Item,
(Optional,Type=Base$List>Iterator,In)Position:P%Pos

Method: RetrieveElementAt

Description: RetrieveElementAt retrieves an element of the array based on it's position.

Options: Public,Method,Void,Platform=All

Parameters:
(Optional,Type=Variant,In)Unknown:P%Unknown,
(Optional,Type=Base$List>Iterator,In)CurrentPosition:P%Iter,
(Optional,Type=Numeric,In)Position:P%Pos

Method: RetrieveFirstElement

Description: RetrieveFirstElement retrieves the first element of the list.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RetrieveLastElement

Description: RetrieveLastElement retrieves the Last element of the list.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: AllowsDuplicates

Description: AllowsDuplicates is a read-only property that indicates whether or not duplicates are allowed in the given collection.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: CurrentPosition

Description: The current position of a List object is a reflection of the sum of all operations to the list. Each operation-- insert, retrieve, etc., changes the current position. Because of timing considerations caused by multiple objects accessing the same list, the current position cannot be relied upon for external operations. It is better to use iterator objects instead of position since it is the responsibility of the iterator object to be able to track the given object in the list regardless of its position.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IsOrdered

Description: IsOrdered returns a bollean value. A 0 is returned If the collection is unordered. A 1 is returned If the collection is ordered. Lists are always ordered and te property will return a 1.

"Ordered" is defined to reflect the constancy of retrieval. For example, if an iterator is used to retrieve the next element from a specified position in an array, it can be assumed that the same data element will be presented each time the call is made-- provided there are no inserts or removes done between the calls.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

I%CurrentPosId

Description: Contains the Id of the item pointed to by the I%CurrentPosition variable.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%CurrentPosition

Description: The current position for element insertion. (Elements of a list are numbered 1 through n.) This variable is exposed as a property (Value only) and is set as a side effect of insert, remove, replace, retrieve operations executed on the list. An insert sets the current position to after the newly inserted element. A remove operation sets the current position to the number preceding the element removed, unless the element removed was the first, in which case the current position is set to 1. Replace and retrieve ops both set current position to the number of the element replaced or retrieved. The InsertElement operation inserts an element at the current position within the list. It is semantically an InsertElementAt operation.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%Head

Description: The position of the first element in the collection. (Elements in a list are numbered from 1 through n.)

Inheritable: Yes
Initialization: Static

Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%Tail

Description: The position of the tail element in the collection. (Elements in a list are numbered from 1 through n where n is the Tail.)

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$List>Iterator

Description: A ListIterator object contains iterator functions that operate specifically on a List collection. It can only be created through the CreateIterator method in the Primary interface of the List object.

Type: Concrete
Superclasses: Collection>Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the List iterator.

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Required,Type=Numeric,In)Position:P%Pos,
(Required,Type=Variant,In)Previous:P%PrevId,
(Required,Type=Variant,In)Next:P%NextId

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the List collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the List collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All,Abstract

Parameters: None

Method: Next

Description: Next returns the next element in the List collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (RequiredRequired,System,Type=Variant,In)Order:P%Order

Instance Variables:

Class: Base$ListIterator

Description: A ListIterator object contains iterator functions that operate specifically on a List collection. It can only be created through the CreateIterator method in the Primary interface of the List object.

Type: Concrete
Superclasses: Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the List iterator.

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Required,Type=Numeric,In)Position:P%Pos,
(Required,Type=Variant,In)Previous:P%PrevId,
(Required,Type=Variant,In)Next:P%NextId

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the List collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the List collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: Next

Description: Next returns the next element in the List collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (RequiredRequired,System,Type=Variant,In)Order:P%Order

Instance Variables:

Class: Base$Log

Description: A Log collection is a collection that is ordered by time. Each element entered into the log collection has an explicit or implicit time stamp associated with it. This serves to order the element in the collection. Time stamps can be included with the element when the element is inserted or the time stamp can be unspecified, in which case the current time is used.

Type: Concrete

Superclasses: Collection

Interface: Factory

Method: CREATE

Description: CREATE creates the log object. P%Data is an optional list of log items to be inserted at creation. P%Data itself contains the number of entries in the P%Data array. Note that data elements may or may not be passed in. The log is created regardless.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DestroyAll

Description: The DestroyAll method is an internal method that destroys all elements of the collection. It is used to clear the collection of elements for reuse.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Method: ContainsElement

Description: Determines whether the collection contains a specific element.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)o:P%Item,
(Optional,Type=String,In)Item:P%Item

Method: Copy

Description: Copies the elements of an Log collection into a target collection. The target collection may be any collection type compatible with the Log collection.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)Log:P%Target

Method: CreateIterator

Description: Creates a Log iterator object that is used to traverse an Log collection. Execution of this method is the only way an iterator can be created.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InsertElement

Description: InsertElement inserts a given object into the Log at the current position. The cardinality of the object is increased. Insertion is automatically ordered based on the associated log time.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Base$TimeStamp,In)TimeStamp:P%TS

Method: InsertElementAt

Description: InsertElementAt inserts the data element in the log at the position specified by the iterator. The position is defined to be a time. Consequently, if an item is inserted at a particular time that results from a traversal is a position defined by days and seconds.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Base$Log>Iterator,In)ObjectCurrentPosition:P%Iter,
(Required,Type=Variant,In)Item:P%Item

Method: RemoveAll

Description: Removes all the Log items and sets the cardinality to 0. If AutoDestroy is true, all objects pointed to will be destroyed as well.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RemoveAllOccurrences

Description : Removes all items from a collection that allows duplicates. This abstract method is overriden in subclasses that allow duplicates. For those collections that do not allow duplicates, this method calls RemoveElement.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)o:P%Item,
(Required,Type=Variant,In)Item:P%Item

Method: RemoveElement

Description: Removes the specified item from the collection. RemoveElement finds the element by its value and removes it. Note, if there are two identical elements in the array, only the first one is removed.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Item

Method: RemoveElementAt

Description: RemoveElementAt positionally removes an object from the Log. An iteration object must be used to specify the position.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Base$Array>Iterator,In)CurrentPosition:P%Iter

Method: RetrieveElementAt

Description: RetrieveElementAt retrieves an element of the array based on it's position.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Log>Iterator,In)CurrentPosition:P%Iter

Method: RetrieveTimeStamp

Definition: RetrieveTimeStamp returns the time stamp of an element in the log, given the item. The log is searched for the item until it is found and the time stamp is returned.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Item

Property: AllowsDuplicates

Description: AllowsDuplicates is a read-only property that indicates whether or not duplicates are allowed in the given collection.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IsOrdered

Description: IsOrdered returns a bollean value. A 0 is returned If the collection is unordered. A 1 is returned If the collection is ordered. Logs are always ordered and the property will retrun a 1.

"Ordered" is defined to reflect the constancy of retrieval. For example, if an iterator is used to retrieve the next element from a specified position in an array, it can be assumed

that the same data element will be presented each time the call is made-- provided there are no inserts or removes done between the calls.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

Class: Base$Log>Iterator

Description: A LogIterator object contains iterator functions that operate specifically on a Log collection. It can only be created through the CreateIterator method in the Primary interface of the Log object.

Type: Concrete
Superclasses: Collection>Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the Log iterator.

Method: CREATE

Description: CREATE command for log iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the appropriate collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Property: CurrentPosition

Description: CurrentPosition returns the internal number representing the current iterator's position in the collection object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given Log collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the Log collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the Log collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: MoveTo

Description: MoveTo allows the position of the iterator to the approximation of the value of a passed in timestamp. If the log contains a record of the exact same time stamp, then the iterator is positioned to that log entry. If not, the log entry that immediately follows the passed in timestamp is used for the positioning of the iterator. If the following log entry is beyond the end of the log, the last log entry is presented effectively executing the Last method.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Base$TimeStamp,In)TimeStamp:P%TimeStamp

Method: Next

Description: Next returns the next element in the Log collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Reset

Description: Reset brings the iterator back to the initial state. Iteration order is maintained.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Current

Description:
This property allows the retrieving of the value of the timestamp being used by the iterator. This, in effect, is the means getting the position of the iterator.
Access: value
Side effect: none
Example:
S ctime=I%itrtr.Current
Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (RequiredRequired,System,Type=Variant,In)Order:P%Order

Instance Variables:

I%LogC

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%LogD

Inheritable: Yes

Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%LogS

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$LogIterator

Description: A LogIterator object contains iterator functions that operate specifically on a Log collection. It can only be created through the CreateIterator method in the Primary interface of the Log object.

Type: Concrete
Superclasses: Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the Log iterator.

Method: CREATE

Description: CREATE command for log iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the appropriate collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Property: CurrentPosition

Description: CurrentPosition returns the internal number representing the current iterator's position in the collection object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given Log collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the Log collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the Log collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: MoveTo

Description: MoveTo allows the position of the iterator to the approximation of the value of a passed in timestamp. If the log contains a record of the exact same time stamp, then the iterator is positioned to that log entry. If not, the log entry that immediately follows the passed in timestamp is used for the positioning of the iterator. If the following log entry is beyond the end of the log, the last log entry is presented effectively executing the Last method.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Base$TimeStamp,In)TimeStamp:P%TimeStamp

Method: Next

Description: Next returns the next element in the Log collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Reset

Description: Reset brings the iterator back to the initial state. Iteration order is maintained.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Current

Description:
This property allows the retrieving of the value of the timestamp being used by the iterator. This, in effect, is the means getting the position of the iterator.
Access: value
Side effect: none
Example:
S ctime=I%itrtr.Current
Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (RequiredRequired,System,Type=Variant,In)Order:P%Order

Instance Variables:

I%LogC

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%LogD

Inheritable: Yes

Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%LogS

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$MVariable

Description: Astract class that defines the operations for an M array variable.

Type: Concrete
Superclasses: Immutable

Interface: Factory

Description: Interface that contains all constructor and destructor operations.

Method: CREATE

Description: Abstract constructor that is Invoked at the time of MVariable object creation. Creates the $Zvirdata variable for the virtual object based on the parameter passed in.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$MVariable,In)Glvn:P%Glvn

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the MVariable class.

Method: Copy

Description: Removes the target location and merges the current object into it.

Options: Public,Method,Void,Platform=All

Parameters: (Required,Type=Base$MVariable,In)Glvn:P%Glvn

Method: Delete

Description: Deletes the value (sets to null) of the MVariable.

Options: Public,Method,Void
Parameters: None

Method: Kill

Description: Kills the M variable name and value.

Options: Public,Method,Void
Parameters: None

Method: Merge

Description: Merges the array reference with the current MVariable array reference.

Options: Public,Method,Void
Parameters: None

Method: Move

Description: Merge to the target (Destination) and delete the source.

Options: Public,Method,Void
Parameters: None

Property: Data

Description: Returns the $Data for this MVariable

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: GLVN

Description: Returns the MVariable reference.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Value

Description: Returns the value bound to the MVariable name.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void
Parameters: None

Class: Base$MVariableStream

Description: Provides a generic IO interface to an M array based on the M Open, Use, Read, Write and Close commands. Abstracts a global as a stream. A simple implementation that will model a given glvn as a stream. Each "line" in the stream is stored. Mixed Read/Write Mode is not suggested.

Type: Concrete
Superclasses: Stream

Interface: Factory

Description: Interface the contains constructors and desctructors.

Method: CREATE

Description: Constructor that initialized the I%Root variable to the M array reference at object instantiation time.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)I%Root

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: This is the public interface for all abstract MVariableStream services.

Method: Read

Description: Reads the specified number of characters from the stream.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)Len:P%Len

Method: ReadLine

Description: Reads a ine from the Stream (Delimited by cr/lf)

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Reset

Description: Reads the specified number of characters from the stream.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Write

Description: Writes data to the Stream.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Text:P%Text

Method: WriteLine

Description: Writes to a Stream apennding a cr/lf.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Line:P%Line

Instance Variables:

I%Blocks

Description: Internal count of the number of $Maxstr blocks are present.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

l%CurBlock

Description: Identifies the "current" block of ESI$Text from which the stream reader will access data.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1
Creation Options: Manually Maintained=No, Dependent=Yes

l%CurChar

Description: Identifies the "current" character within the current block of ESI$Text from which the stream reader will access data.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1
Creation Options: Manually Maintained=No, Dependent=Yes

l%LastLen

Description: The size of the final text block.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

l%Length

Description: The total number of characters in the text.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

l%Root

Description: The global root to use for the stream.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Map

Description: A Map collection manages a set of ordered pairs, in which each key is associated with exactly one value. A Map permits you to easily find the value associated

with a particular key. This collection is similar to MultiMap, which permits more than one value per key.

Type: Concrete
Superclasses: KeyedCollection

Interface: Factory

Method: CREATE

Description: CREATE creates the basic collection. Input can be one or more objects to be inserted. The different subclasses of collection objects handles any order or key storage necessary to the object. Note that data elements may or may not be passed in. The collection is created regardless.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DESTROY

Description: DESTROY destroys the collection. All iterators are removed.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DestroyAll

Description: The DestroyAll method is an internal method that destroys all elements of the collection. It is used to clear the collection of elements for reuse.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Event: ElementAdded

Description: The ElementAdded event is thrown when an element is added to the collection. The event passes the collection IOD, the event name, the key and value added to the watcher.

Callback Documentation:
EleAdded(T%OID,T%Event,T%Key,T%Value) ; Key/Value pair added to a map collection
; T%OID - the collection object in which an element was added.
; T%Event - the name of the event "ElementAdded".
; T%Key - the Key added.
; T%Value - the value added with the key.

Event: ElementRemoved

Description: The ElementRemoved event is thrown when an element is removed from the collection. The event passes the collection OID, the event name, the key and value that has been removed to the watcher.

Callback Documentation:
EleRem(T%OID,T%Event,T%Key,T%Value) ; Key/Value pair removed from a map collection
; T%OID - the collection object in which an element was removed.
; T%Event - the name of the event "ElementRemoved".
; T%Key - the key removed.
; T%Value - the value removed.

Method: ContainsElement

Description: Determines whether the collection contains a specific element.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)P%Item

Method: ContainsKey

Description: ContainsKey returns whether or not a key exists in the Map for any item. This is useful to detect if a particular key is in use in the Map.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%Key

Method: Copy

Description: Copies the elements of an Map collection into a target collection. The target collection may be any collection type compatible with the Map collection.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)Map:P%Target

Method: CreateIterator

Description: Creates a Map iterator object that is used to traverse an Map collection. Execution of this method is the only way an iterator can be created.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DestroyAll

Description: Destroys all the elements in the Map object.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InsertElement

Description: InsertElement inserts a new item into the Map.
Note:
1) a null key can be used.
2) a key can only be used in the map once.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Key:P%Key,
(Required,Type=Variant,In)Value:P%Item=""

Method: InsertElementAt

Description: InsertElementAt inserts the data element in the collection at a specified position.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Required,Type=Variant,In)pos:P%Key

Method: Merge

Description: Merges the contents of the current Map into a second Map.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)Map:P%MapCopy

Method: RemoveAll

Description: Removes all the Map items and sets the cardinality to 0. If AutoDestroy is true, all objects pointed to will be destroyed as well.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RemoveElement

Description: Removes the specified item from the collection. RemoveElement finds the element by its value and removes it. Note, if there are two identical elements in the array, only the first one is removed.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Key

Method: RemoveElementAt

Description: RemoveElementAt positionally removes an item from the Map. An iteration object or a key must be used to specify the position. If the iterator is bad, a null key is used.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Variant,In)Key:P%Key="",
(Optional,Type=Base$Map>Iterator,In)CurrentPosition:P%Iter

Method: ReplaceElementAt

Description: Replaces the element pointed to by the position iterator with the item passed in.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Variant,In)Key:P%Key,
(Optional,Type=Base$Map>Iterator,In)Itr:P%Iter

Method: RetrieveElement

Description: RetrieveElement retrieves the contents of the Map as positionally described. Calling this method without any parameters returns the item presently at the null key position.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Key

Method: RetrieveElementAt

Description: RetrieveElementAt retrieves an element of the array based on it's position.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Variant,In)key:P%Key,
(Optional,Type=Base$Map>Iterator,In)CurrentPosition:P%Iter

Method: RetrieveElementsByKey

Description: RetrieveElementsByKey returns a list object of all of the items that satisfy a given key. The first position of the list contains the key passed in. Note: Since Maps cannot have redundant keys, the list will not contain more than one object (plus the key).

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%Key

Method: RetrieveElementsByPattern

Description: RetrieveElementsByPattern returns a list collection filled items whose key satisfy a given M pattern match.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Pattern:P%Pattern

Method: SetElement

Allows an element in the map to be set by Key. Both the Key and the new value are required parameters. The Key need to already exist to be set.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Key:P%Key,
(Required,Type=Variant,In)Item:P%Item

Property: AllowsDuplicates

Description: AllowsDuplicates is a read-only property that indicates whether or not duplicates are allowed in the given collection.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IsOrdered

Description: IsOrdered returns a bollean value. A 0 is returned If the collection is unordered. A 1 is returned If the collection is ordered. Maps are always ordered and will return a 1.

"Ordered" is defined to reflect the constancy of retrieval. For example, if an iterator is used to retrieve the next element from a specified position in an array, it can be assumed that the same data element will be presented each time the call is made-- provided there are no inserts or removes done between the calls.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

I%IL

Description: A builtin array for the items inserted in the collection.

Structure:

I%IL(ShortKey)=Item
I%IL(ShortKey,0) = Level Count
I%IL(ShortKey,n)= Key
I%IL(ShortKey,n,0) = Item

Where:
ShortKey is the Key Truncated to 60 Characters.
Item is the Data Item.

Inheritable: Yes
Initialization: Static
Binding: BuiltInString
Creation Options: Manually Maintained=No, Dependent=Yes

I%NIL

Description: A builtin variable used to store the single item whose Key is NULL.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Map>Iterator

Description: A MapIterator object contains iterator functions that operate specifically on a Map collection. It can only be created through the CreateIterator method in the Primary interface of the Map object.

Type: Concrete
Superclasses: KeyedCollection>Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the Map iterator.

Method: CREATE

DESCRIPTION: CREATE command for Map iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the Map collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: DESTROY

DESCRIPTION:
Destroys the particular iterator associated with the given collection object.
SIDE EFFECTS:
Notifies collection object about iterator removal.
RETURN VALUE:
None
EXAMPLE:
Destroy A%MyIterator
Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Method responsible for initializing class variables. Be very careful when editing this code!
Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Method responsible for initializing system variables. Be very careful when editing this code!
Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then

the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%Key

Method: SuperSort

Description: This function returns the 'next' extended key relative to the current direction (compared to the present extended key). If no 'next' key is found, the function returns null.

Note: Do not call this method directly. This method is used internally by the Next method for the Map Iterator and is not intended for external use!

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Numeric,In)P%Type,
(Optional,Type=Variant,In)P%Start=$$^VESO1("CurrentKey"),
(Optional,Type=Numeric,In)P%Dir=$$^VESO1("NumOrder")

Property: CurrentPosition

Description: CurrentPosition returns the internal number representing the current iterator's position in the collection object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given Map collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the Map collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the Map collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Next

Description: Next returns the next element in the Map collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Reset

Description: Reset brings the iterator back to the initial state. Iteration order is maintained.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: CurrentItem

Description: This property retrieves the value of the current item found by the last iterator operation.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: CurrentKey

Description: The CurrentKey property assigns and retrieves the value of the key being used by the iterator.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: DeadSpot

DESCRIPTION:
Tells the user whether the CurrentKeyItemPosition properties are set to existing values or not.
ACCESS:
Value.
SIDE EFFECTS:
None.

EXAMPLES:
Value:
S A%Result=I%MyIterator.DeadSpot
Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Truth:P%Truth

Property: EndKey

Description: The EndKey defines the end of the traverse, which can be at either end of the dictionary depending on the traverse order. NOTE: Changing the EndKey value does NOT change the current key position of the iterator. This is true even if changing the EndKey value puts the position beyond the end of the range.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Order:P%Order

Property: StartKey

Description: The StartKey defines the beginning of the key traversal, which can be at either end of the dictionary depending on the traverse order.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Instance Variables:

I%CurrentKey

Description: The last key that the iterator pointed to. The position need not exist.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%DeadSpot

Description: Boolean value representing the actual existence of CurrentKey\Item\Position.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%EndKey

Description: The ending key (which may or may not exist) that iteration will end on.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%NumOrder

Description: A numeric value representing the current direction (1=foward,-1=backward). The default is 1.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%StartKey

Description: The starting key (which may or may not exist) that iteration will begin on.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$MapIterator

Description: A MapIterator object contains iterator functions that operate specifically on a Map collection. It can only be created through the CreateIterator method in the Primary interface of the Map object.

Type: Concrete
Superclasses: KeyedIterator

Interface: Factory

Description: Contains all constructor and destructor services for the Map iterator.

Method: CREATE

DESCRIPTION: CREATE command for Map iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the Map collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: DESTROY

DESCRIPTION:
Destroys the particular iterator associated with the given collection object.
SIDE EFFECTS:
Notifies collection object about iterator removal.
RETURN VALUE:
None
EXAMPLE:
Destroy A%MyIterator
Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Method responsible for initializing class variables. Be very careful when editing this code!
Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Method responsible for initializing system variables. Be very careful when editing this code!
Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%Key

Method: SuperSort

Description: This function returns the 'next' extended key relative to the current direction (compared to the present extended key). If no 'next' key is found, the function returns null.

Note: Do not call this method directly. This method is used internally by the Next method for the Map Iterator and is not intended for external use!

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Numeric,In)P%Type,
(Optional,Type=Variant,In)P%Start=$$^VESO1("CurrentKey"),
(Optional,Type=Numeric,In)P%Dir=$$^VESO1("NumOrder")

Property: CurrentPosition

Description: CurrentPosition returns the internal number representing the current iterator's position in the collection object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given Map collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the Map collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the Map collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Next

Description: Next returns the next element in the Map collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Reset

Description: Reset brings the iterator back to the initial state. Iteration order is maintained.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: CurrentItem

Description: This property retrieves the value of the current item found by the last iterator operation.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: CurrentKey

Description: The CurrentKey property assigns and retrieves the value of the key being used by the iterator.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: DeadSpot

DESCRIPTION:
Tells the user whether the CurrentKeyItemPosition properties are set to existing values or not.
ACCESS:
Value.
SIDE EFFECTS:

None.

EXAMPLES:
Value:
S A%Result=I%MyIterator.DeadSpot
Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Truth:P%Truth

Property: EndKey

Description: The EndKey defines the end of the traverse, which can be at either end of the dictionary depending on the traverse order. NOTE: Changing the EndKey value does NOT change the current key position of the iterator. This is true even if changing the EndKey value puts the position beyond the end of the range.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Order:P%Order

Property: StartKey

Description: The StartKey defines the beginning of the key traversal, which can be at either end of the dictionary depending on the traverse order.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Instance Variables:

I%CurrentKey

Description: The last key that the iterator pointed to. The position need not exist.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%DeadSpot

Description: Boolean value representing the actual existence of CurrentKey\Item\Position.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%EndKey

Description: The ending key (which may or may not exist) that iteration will end on.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%NumOrder

Description: A numeric value representing the current direction (1=foward,-1=backward). The default is 1.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%StartKey

Description: The starting key (which may or may not exist) that iteration will begin on.

Inheritable: Yes
Initialization: Static
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$MultiMap

Description: A MultiMap collection manages a set of ordered pairs in which each key is associated with one or more values. A MultiMap permits you to easily find all of the values associated with a particular key. This collection is similar to Map, which only permits one value per key.

The keys in the collection are ordered according to the normal sorting sequence. If multiple values contain the same keys, then no particular ordering among them is guaranteed.

Type: Concrete
Superclasses: KeyedCollection

Interface: Factory

Method: CREATE

Description: CREATE creates the basic collection. P%Data is a list of element to be inserted. P%Data itself contains the number of entries in the P%Data array. The different subclasses of collection objects handles any order or key storage necessary to the object. Note that data elements may or may not be passed in. The collection is created regardless.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DestroyAll

Description: The DestroyAll method is an internal method that destroys all elements of the collection. It is used to clear the collection of elements for reuse.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Interface: Primary

Event: ElementAdded

Description: The ElementAdded event is thrown when an element is added to the collection. The event passes the collection IOD, the event name, the key and value added to the watcher.

Callback Documentation:

```
EleAdded(T%OID,T%Event,T%Key,T%Value)      ; Key/Value pair added to a multimap collection
        ; T%OID - the collection object in which an element was added.
        ; T%Event - the name of the event "ElementAdded".
        ; T%Key - the Key added.
        ; T%Value - the value added with the key.
        Q
```

Event: ElementRemoved

Description: The ElementRemoved event is thrown when an element is removed from the collection. The event passes the collection OID, the event name, the key and value that has been removed to the watcher.

Callback Documentation:

```
EleRem(T%OID,T%Event,T%Key,T%Value)      ; Key/Value pair removed from a multi-map collection
        ; T%OID - the collection object in which an element was removed.
        ; T%Event - the name of the event "ElementRemoved".
        ; T%Key - the key removed.
        ; T%Value - the value removed.
```

Method: ContainsElement

Description: Determines whether the collection contains a specific element.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)P%Item

Method: ContainsKey

Description: Determines if the MultiMap collection contains a specific key.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%Key

Method: Copy

Description: Copies the elements of an MultiMap collection into a target collection. The target collection may be any collection type compatible with the MultiMap collection.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)P%Target

Method: CreateIterator

Description: Creates an MultiMap iterator object that is used to traverse an MultiMap collection. Execution of this method is the only way an iterator can be created.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: CreateKeyIterator

Description: Creates an special purpose MultiMap iterator object that is used to traverse a MultiMap collection by key. Execution of this method is the only way an iterator can be created.
Options: Public,Method,Void
Parameters: None

Method: InsertElement

Description: Inserts an Item into the MultiMap object. No key implies an empty key. Note: Trying to insert a key/element pair that already exists results in failure. Use 'ReplaceElement' for that purpose.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Key:P%Key="",
(Required,Type=Variant,In)Value:P%Item

Method: RemoveAll

Description: Removes all the MultiMap items and sets the cardinality to 0. If AutoDestroy is true, all objects pointed to will be destroyed as well.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RemoveElement

Description: Removes the specified item from the collection. RemoveElement finds the element by its value and removes it. Note, if there are two identical elements in the array, only the first one is removed.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Key

Method: RemoveElementAt

Description: RemoveElementAt positionally removes an item from the MultiMap. An iteration object or a key must be used to specify the position. If the iterator is bad, a null key is used. Note: Passing in a key will remove all references to the given key while passing in an iterator will remove only the item that the iterator currently points to.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)P%KEYorIT

Method: ReplaceElementAt

Description: Replaces the element pointed to by the position iterator with the item passed in.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Base$MultiMap>Iterator,In)Iterator:P%Iterator,
(Required,Type=Variant,In)Item:P%NewItem,
(Required,Type=Variant,In)Key:P%Key

Method: RetrieveElement

Description: RetrieveElement retrieves the contents of the MultiMap as described by the user supplied key. Calling this method without any parameters returns the item(s) presently stored at the null key position.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)P%Key

Method: RetrieveElementAt

Description: RetrieveElementAt retrieves an element of the array based on it's position.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)P%KEYorIT

Method: RetrieveElementsByKey

Description: RetrieveElementsByKey returns a list object of all of the items that satisfy a given key. The first position of the list contains the key passed in. Note: Since Maps cannot have redundant keys, the list will not contain more than one object (plus the key).

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%KEYorIT

Method: RetrieveElementsByPattern

Description: RetrieveElementsByPattern returns a list collection filled items whose key satisfy a given M pattern match.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Pattern:P%Pattern

Method: RetrieveKeysByPattern

Description: Returns a list collection filled with keys whose item(s) satisfy a given M pattern match.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Pattern:P%Pattern

Property: AllowsDuplicates

Description: AllowsDuplicates is a read-only property that indicates whether or not duplicates are allowed in the given collection.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IsOrdered

Description: IsOrdered returns a bollean value. A 0 is returned If the collection is unordered. A 1 is returned If the collection is ordered. As all MultiMaps are ordered, this function will return true for all cases. The Ordering of the items within the multimap is by the key of the objects inserted.

"Ordered" is defined to reflect the constancy of retrieval. For example, if an iterator is used to retrieve the next element from a specified position in an array, it can be assumed that the same data element will be presented each time the call is made-- provided there are no inserts or removes done between the calls.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

I%IL

Description: A builtin array for the items inserted in the collection.

Normal Key Structure
I%IL(Key,0)=Index - Next available item node2
I%IL(Key,Node2)=Item - Any given item for specific key

Extended Key Structure
I%IL(EKey,0)=Index
tab - Next available extended key node2
I%IL(EKey,Node2)=Full Key - Any given full key
I%IL(EKey,Node2,0)=Index - Next available item node3
I%IL(EKey,Node2,Node3)=Item - Any given item for specific key
---
Null Key Structure
I%NIL(0)=Index - Next available item node
I%NIL(Node1)=Item - Any given item with an empty key value

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%NIL

Description: Used to store the item(s) whose Key is NULL.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$MultiMap>Iterator

Description: A MultipMapIterator object contains iterator functions that operate specifically on a MultiMap collection. It can only be created through the CreateIterator method in the Primary interface of the MultiMap object.

Type: Concrete
Superclasses: KeyedCollection>Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the MultiMap iterator.

Method: CREATE

Description: General CREATE command for iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the appropriate collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%Key

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given MultiMap collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the MultiMap collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Next

Description: Next returns the next element in the MultiMap collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: CurrentItem

Description: This property retrieves the value of the current item found by the last iterator operation.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Item:P%Item

Property: CurrentKey

Description: The CurrentKey property assigns and retrieves the value of the key being used by the iterator.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All

Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: DeadSpot

DESCRIPTION:
Tells the user whether the CurrentKeyItemPosition properties are set to existing values or not.
ACCESS:
Value.
SIDE EFFECTS:
None.

EXAMPLES:
Value:
S A%Result=I%MyIterator.DeadSpot
Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Truth:P%Truth

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Order:P%Order

Instance Variables:

I%CurrentKey

Description: The last key that the iterator pointed to. The position need not exist.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%DeadSpot

Description: Boolean value representing the actual existence of CurrentKey\Item\Position.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%EndKey

Description: The ending key (which may or may not exist) that iteration will end on.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%MarkerOne

Description: Used internally to mark iteration paths through keys.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%MarkerTwo

Description: Used internally to mark iteration paths through keys.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%NumOrder

Description: A numeric value representing the current direction (1=foward,-1=backward). The default is 1.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%StartKey

Description: The starting key (which may or may not exist) that iteration will begin on.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$MultiMapIterator

Description: A MultipMapIterator object contains iterator functions that operate specifically on a MultiMap collection. It can only be created through the CreateIterator method in the Primary interface of the MultiMap object.

Type: Concrete
Superclasses: KeyedIterator

Interface: Factory

Description: Contains all constructor and destructor services for the MultiMap iterator.

Method: CREATE

Description: General CREATE command for iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the appropriate collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void

Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)Key:P%Key

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given MultiMap collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the MultiMap collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Next

Description: Next returns the next element in the MultiMap collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: CurrentItem

Description: This property retrieves the value of the current item found by the last iterator operation.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Item:P%Item

Property: CurrentKey

Description: The CurrentKey property assigns and retrieves the value of the key being used by the iterator.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Key:P%Key

Property: DeadSpot

DESCRIPTION:
Tells the user whether the CurrentKeyItemPosition properties are set to existing values or not.
ACCESS:
Value.
SIDE EFFECTS:
None.

EXAMPLES:

Value:
S A%Result=I%MyIterator.DeadSpot
Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Truth:P%Truth

Property: IterationOrder

Description: IterationOrder implements the Assign and Value accessors that set and retrieve the current order of the iterator respectively.
F = Forward
1 = Forward
B = Backward
-1 = Backward

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Method,Void,Platform=All
Parameters: (Required,System,Type=Variant,In)Order:P%Order

Instance Variables:

I%CurrentKey

Description: The last key that the iterator pointed to. The position need not exist.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%DeadSpot

Description: Boolean value representing the actual existence of CurrentKey\Item\Position.

Inheritable: Yes

Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%EndKey

Description: The ending key (which may or may not exist) that iteration will end on.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%MarkerOne

Description: Used internally to mark iteration paths through keys.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%MarkerTwo

Description: Used internally to mark iteration paths through keys.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%NumOrder

Description: A numeric value representing the current direction (1=foward,-1=backward). The default is 1.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%StartKey

Description: The starting key (which may or may not exist) that iteration will begin on.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$MultiMapKeyIterator

Description: This class inherits from MultiMapIterator. This special purpose iterator traverses the MultiMap by key. The First, Last and Next operations will return a multimap collection of the contained within the current key.

Type: Concrete
Superclasses: MultiMapIterator

Interface: Factory

Description: Contains all constructor and destructor services for the MultiMapKey iterator.

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: First

Description: First returns the first element of the given MultiMapKey collection based on the current direction of the iterator. If the direction is backwards, the "first" element is the last element.
Options: Public,Method,Void,Platform=All
Parameters: None

Method: Last

Description: Last returns the last element of the MultiMapKey collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.
Options: Public,Method,Void,Platform=All
Parameters: None

Method: More

Description: More returns a truth value indicating more or no more elements in the more collection based on the current iteration direction.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Numeric,In)Position:P%Number

Method: Next

Description: Next returns the next element in the MultiMapKey collection. Null is returned if the end of the collection is reached.
Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

Class: Base$NameValuePair

Description: The NameValuePair immutable is used to hold a Name and its Value as a virtual object.

Type: Concrete
Superclasses: Immutable

Interface: Factory

Description: The NameValuePair interface that holds the constructors and destructors.

Method: CREATE

Description: Constructor that creates the $ZVIRDATA special variable from the Name and Value.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=String,In)Name:P%Name,
(Required,Type=String,In)Value:P%Value

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the NameValuePair class.

Property: Name

Description: Returns the Name of the Name Value construct.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Value

Description: Returns the value for the Name Value construct.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Class: Base$NewNamePool

Description: An instance of the Name Pool class acts as a symbol table that is addressable via symbol scope N%name where name can be the M array structure. NamePool instances can be linked into hierarchies that support inheritance and name overriding. NamePool objects are sharable between objects and are a convienent way of sharing information.

Usage: NamePool variables are addressed via the N%name variable scope. They have a full reference structure of N%(Pool OID)name where N% identifies it as a name pool variable, Pool OID is the OID of the pool object that the N%name variable reside in and name is the actual variable name. To avoid embedding the (Pool OID) within the variable, a special variable $POOL can be used to hold the Pool OID. It then acts as a default pool pointer within the execution scope of the current method.

Examples:

1) Set N%(T%PooOID)Person("Name")="Doe, John"

2) Set $Pool=T%PoolOID
   Set N%Person("Name")="Doe, John"

Type: Concrete

Interface: Factory

Description: Interface the contains constructors and desctructors for the NewNamePool class.

Method: CREATE

Description: Cosntructor that intialized the NamePool pointer.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: CreateDescendant

Description: Creates a descentant name pool object.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DESTROY

Description: Messages the ancestor (if it exists) to unlink this object from its descendant list.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DestroySymbols

Description: Causes all Top level symbols in the name pool to be destroyed. Normally a name pool is assumed not to own the symbols contained with in it, by calling this method prior to destroying the name pool the symbols contained with in the name pool may be destroyed.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: This is the public interface for all NewNamePool services.

Method: AncestorLink

Description: Links the current name pool object to its parent. This is called by the CreateDescendant method in the Factory interface to link the creator to the descendant as its ancestor.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$NewNamePool,In)Parent:P%Parent

Method: AncestorUnlink

Description: Unlink the current name pool from its ancestor if it exists.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DescendantLink

Description: Links a name pool object to the current name pool after making some validity checks.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Base$NewNamePool,In)Parent:P%To

Method: DescendantUnlink

Description: Unlinks a descendant from the current name pool.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: ReportLink

Description: Returns the ancestor's OID or null if it does not exist.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Name

Description: Exposes the I%Name instance variable via property accessors: Assign, Value, Kill, $Get and $Data
Parameters: See each property accessor.
Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: $Get

Options: Public,Property_Value,Void,Platform=All
Parameters: (Required,System,Type=Numeric,In)Default:P%Default=""

Accessor: $Data

Options: Public,Property_Data,Void,Platform=All
Parameters: None

Accessor: Kill

Options: Public,Property_Kill,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All
Parameters: (Required,System,Type=String,In)Name:P%Nam

Interface: VariableFactory

Description: This is the variable initialization interface for the NewNamePool class.

Method: Descendants

Options: Public,Method,Void
Parameters: None

Method: Name

Options: Public,Method,Void
Parameters: None

Instance Variables:

I%Ancestor

Description: This variable holds a pointer (OID) of the current name pools parent.

Inheritable: Yes
Initialization: Static
Binding: Base$NewNamePool
Creation Options: Manually Maintained=No, Dependent=No
Creation Keyword: CHILD=1
Creation Parameters:

I%Descendants

Description: The array variable holds a list of descendants of the current object with the structure Descendants(OID)="".

Inheritable: Yes
Initialization: Initialized
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%Name

Description: Name contains the name of the pool object.

Inheritable: Yes
Initialization: Initialized
Binding: Expression=$s(I%Ancestor'="":">"_I%Ancestor.tName,1:"Unnamed Pool")
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$OleObject

Type: Concrete

Interface: Factory

Method: CREATE

Options: Public,Method,Void
Parameters:
(Optional,Type=String,In)Class:A%OleClass,
(Optional,Type=String,In)File:A%BaseFile

Interface: Primary

Method: Test

Options: Public,Method,Void
Parameters: None

Class: Base$OrCompoundCriteria

A disjunctive compound Criteria, returning TRUE when any of its component criteria are true, and FALSE if all of them are false (or if it does not contain any criteria.)
Type: Concrete
Superclasses: CompoundCriteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Method: Matches

Description: Returns true if ANY of the Criteria are true, false only if ALL are false. Also returns false if there are no criteria.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)o:P%Object

Instance Variables:

Class: Base$PatternCriteria

Description: Implements the pattern matching criteria operations.

Type: Concrete
Superclasses: RelationalCriteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: CREATE

Description: The constructor method is used to create any required structures of the object at creation time. This generally includes all static members, and all relationships.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)Property:I%PatProperty,
(Optional,Type=String,In)Pattern:I%Pattern

Method: DESTROY

Description: Destructor for PatternCriteria class.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Method: Matches

Description: Returns true if the value of the specified object's property matches the specified pattern. Returns false if the property does not match the pattern, or if no pattern is defined.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)o:P%Object

Property: Pattern

Description: A pattern to be applied by the Matches method.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void
Parameters: None

Accessor: Assign

Options: Public,Method,Void
Parameters: (Required,System,Type=String,In)A%Value

Property: Properties

Description: Returns a collection containing the properties affected by the criterion. An object must implement these properties in order to be used by the Matches method for this class.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Property

Description: Returns the property affected by the criteria. This propery will be referenced when an object is passed to the Matches method.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All
Parameters: (Required,System,Type=String,In)Property:P%Prop

Instance Variables:

I%PatProperty

Used by the Property property.
Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%Pattern

Used for the Pattern property.
Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$RangeCriteria

Description: RangeCriteria determines whether a specific property of an object falls between a pre-defined starting and ending value (non-inclusive).

Type: Concrete
Superclasses: Criteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: CREATE

Description: The constructor method is used to create any required structures of the object at creation time. This generally includes all static members, and all relationships.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)Property:I%RngProperty,
(Optional,Type=String,In)Start:I%RngStart,
(Optional,Type=String,In)End:I%RngEnd

Method: DESTROY

Description: Destructor for RangeCriteria classes.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Method: Matches

Description: Returns true if the value of the specified object's range property is between the RangeStart and RangeEnd properties (non-inclusive).

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)o:P%Object

Property: IsRange

Description: Returns true if the criterion is based on range (exposing RangeStart and RangeEnd properties), false if not. Always true for RangeCriteria.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Properties

Description: Returns a collection containing the properties affected by the criterion. An object must implement these properties in order to be used by the Matches method for this class.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Property

Description: Returns the property affected by the criteria. This propery will be referenced when an object is passed to the Matches method.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All
Parameters: (Required,System,Type=String,In)Property:P%Prop

Property: RangeEnd

Description: Sets and returns the range's non-inclusive ending value.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All
Parameters: (Required,System,Type=String,In)Value:P%Value

Property: RangeStart

Description: Sets and returns the range's non-inclusive starting value.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All
Parameters: (Required,System,Type=String,In)Value:P%Value

Instance Variables:

I%RngEnd

Corresponds to the RangeEnd property.
Inheritable: Yes

Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%RngProperty

Used by the RangeProperty property.
Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%RngStart

Used by the RangeStart property.
Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Reader

Description: Abstract class whose subclasses implement readers for particular text structures.

Type: Abstract

Class: Base$RelationalCriteria

Description: Abstract class that support relational criteria subclasses.

Type: Abstract
Superclasses: FilterCriteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: CREATE

Description: Creates a relational criteria object.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)Property:I%Property,
(Optional,Type=String,In)RelationalValue:I%RelationalValue

Method: DESTROY

Description: Abstract destructor for RelationalCriteria class.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Property: Properties

Description: Returns a collection containing the properties affected by the criterion. An object must implement these properties in order to be used by the Matches method for this class.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Property

Description: Returns the property affected by the criteria. This propery will be referenced when an object is passed to the Matches method.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All
Parameters: (Required,System,Type=String,In)Property:P%Prop

Property: Value

Description: The value that will be compared to an object's property by the Matches method.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All

Parameters: None

Accessor: Assign

Options: Public,Property_Assign,Void,Platform=All
Parameters: (Required,System,Type=String,In)Value:P%Val

Instance Variables:

I%Property

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%RelationalValue

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$ServerFileStream

Description: Provides a generic IO interface to a host file based on the M Open, Use, Read, Write and Close commands.

Type: Concrete
Superclasses: Stream

Interface: Factory

Description: Interface the contains constructors and desctructors.

Method: CREATE

Description: Constructor that initialized the I%File variable to the file path name at object instantiation time.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)I%File

Method: DESTROY

Description: Destructor that closes the file at object destruction time if it has not been closed.

Options: Public,Method,Void,UsesIO,Platform=All
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: This is the public interface for all abstract ServerFileStream services.

Method: Close

Description: Closes the file device for the default M implementation.

Options: Public,Method,Void,UsesIO,Platform=All
Parameters: None

Method: Open

Description: Opens the File device for the default M implementation.

Options: Public,Method,Void,UsesIO,Platform=All
Parameters: (Optional,Type=String,In)Mode:I%Mode=""

Method: Read

Description: Reads the specified number of characters from the stream.

Options: Public,Method,Void,UsesIO,Platform=All
Parameters: (Required,Type=Numeric,In)Len:P%Len=0

Method: ReadLine

Description: Reads a ine from the Stream (Delimited by cr/lf)

Options: Public,Method,Void,UsesIO,Platform=All
Parameters: None

Method: Write

Description: Writes data to the Stream.

Options: Public,Method,Void,UsesIO,Platform=All
Parameters: (Required,Type=String,In)Text:P%Text=""

Method: WriteLine

Description: Writes to a Stream apennding a cr/lf.

Options: Public,Method,Void,UsesIO,Platform=All
Parameters: (Required,Type=String,In)Line:P%Line

Instance Variables:

I%File

Description: Contains the path and file name.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%LastChar

Description: Contains the last character read.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%Mode

Description: Contains the mode string used in the open of a device.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Set

Description: A Set collection is an unordered collection of objects. It differs from a bag in that a Set does not allow duplicates. All of the operations available to bags is available to sets.

Type: Concrete
Superclasses: Collection

Interface: Factory

Method: CREATE

Description: CREATE creates the set object. P%Data is a list of element to be inserted. P%Data itself contains the number of entries in the P%Data array. Note that data elements may or may not be passed in. The set is created regardless.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DestroyAll

Description: The DestroyAll method is an internal method that destroys all elements of the collection. It is used to clear the collection of elements for reuse.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Method: ContainsElement

Description: Determines whether the collection contains a specific element.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)o:P%Item,
(Optional,Type=String,In)Item:P%Item

Method: Copy

Description: Copies the elements of an Set collection into a target collection. The target collection may be any collection type compatible with the Set collection.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)Set:P%Target

Method: CreateIterator

Description: Creates a Set iterator object that is used to traverse an Set collection. Execution of this method is the only way an iterator can be created.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Difference

Description: Difference returns a new set object which contains set elements that are neither the set being called nor in the set being submitted. For example, if set A has:

1 onion
1 carrot
1 pea

and set B has:

1 carrot
1 pea

Then the difference set would contain:

1 onion

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Set,In)Set:P%Set

Method: InsertElement

Description: InsertElement inserts a new element into the Set object. No presumption of position can be made as to the location of the element in the set. The cardinality is changed.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Item

Method: Intersection

Description: Intersection returns a new set object which contains set elements that are in both the set being called and in the set being submitted. For example, if set A has:

1 onion
1 carrot
1 pea

and set B has:

1 carrot
1 pea

Then the Intersection set would contain:

1 carrot
1 pea

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Set,In)Set:P%Set

Method: IsProperSubset

Description: IsProperSubset returns true if the passed in set is a proper subset of the set collection object. A proper subset is a set of elements, all of which are contained in the set being compared to, but the two sets are not identical.

Options: Public,Method,Void,Platform=All

Parameters: (Required,Type=Base$Set,In)Set:P%Set

Method: IsProperSuperset

Description: IsProperSuperset returns true if the input set of this method is a proper superset of the object being called. A proper superset is a set that contains all of the elements of the current set and is not equal to the current set. This method is the mirror method of the IsProperSubset method.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Set,In)Set:P%Set

Method: IsSubset

Description: IsSubset returns if the passed in set is a proper subset of the set collection object. A proper subset is a set of elements all of which are contained in the set being compared to but the two sets may be identical.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Set,In)Set:P%Set

Method: IsSuperset

Description: IsSuperset returns if the input set of this method is a superset of the object being called. A superset is a set that contains all of the elements of the current set but may or may not be equal to the current set. This method is the mirror method of the IsSubset method.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Set,In)Set:P%Set

Method: RemoveAll

Description: Removes all the Set items and sets the cardinality to 0. If AutoDestroy is true, all objects pointed to will be destroyed as well.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: RemoveElement

Description: Removes the specified item from the collection. RemoveElement finds the element by its value and removes it. Note, if there are two identical elements in the array, only the first one is removed.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Variant,In)o:P%Item

Method: RemoveElementAt

Description: RemoveElementAt positionally removes an object from the Set. An iteration object must be used to specify the position.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Base$Bag>Iterator,In)CurrentPosition:P%Iter

Method: ReplaceElementAt

Description: Replaces the element pointed to by the position iterator with the item passed in.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)o:P%Item,
(Required,Type=Object,In)CurrentPosition:P%Iter

Method: RetrieveElementAt

Description: RetrieveElementAt retrieves an element of the array based on it's position.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Base$Set>Iterator,In)CurrentPosition:P%Iter

Method: Union

Description: Union returns a new set object which contains set elements that are in the set being called or in the set being submitted or both. For example, if set A has:

1 onion
1 carrot
1 pea

and set B has:

1 carrot
1 pea

Then the Union set would contain:

1 onion
1 carrot
1 pea

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Set,In)Set:P%Set

Property: AllowsDuplicates

Description: AllowsDuplicates is a read-only property that indicates whether or not duplicates are allowed in the given collection.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: IsOrdered

Description: IsOrdered returns a bollean value. A 0 is returned If the collection is unordered. A 1 is returned If the collection is ordered. Sets are always unordered and this property will return a 0.

"Ordered" is defined to reflect the constancy of retrieval. For example, if an iterator is used to retrieve the next element from a specified position in an array, it can be assumed that the same data element will be presented each time the call is made-- provided there are no inserts or removes done between the calls.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Interface: VariableFactory

Method: HasNull

Description: System generated method that initializes the I%HasNull variable when accessed dynamically.

Options: Public,Method,Void
Parameters: None

Instance Variables:

I%HasNull

Description: Used as a switch to indicate that a collection has null keys.

Inheritable: Yes
Initialization: Initialized
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%Items

Description: Used as a xref to store objects that are placed in IL.

Inheritable: Yes
Initialization: Static
Binding: Expression

Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Set>Iterator

Description: A SetIterator object contains iterator functions that operate specifically on a Set collection. It can only be created through the CreateIterator method in the Primary interface of the Set object.

Type: Concrete
Superclasses: Collection>Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the Set iterator.

Method: CREATE

Description: General CREATE command for iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the appropriate collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called.

Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: Last

Description: Last returns the last element of the Set collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Next

Description: Next returns the next element in the Set collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

I%HasNull

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$SetIterator

Description: A SetIterator object contains iterator functions that operate specifically on a Set collection. It can only be created through the CreateIterator method in the Primary interface of the Set object.

Type: Concrete
Superclasses: Iterator

Interface: Factory

Description: Contains all constructor and destructor services for the Set iterator.

Method: CREATE

Description: General CREATE command for iterators. SHOULD NOT BE USED DIRECTLY. Should only be used by the CreateIterator method of the appropriate collection object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)List:I%List

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Contains services that are not permitted in the Primary interface and are used internally be public services.

Method: ElementAdded

Description: ElementAdded is used to handle the circumstance of elements being added in an unordered collection. For example, if an element is inserted into a bag object, then the iterator at some point would have to encounter the element and return it. For this purpose, whenever an element is added to the collection object, this method is called. Newly added elements will "float" to the bottom of processing. Since the collections concerned are unordered collections this is not a problem.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of adding an element to the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Method: ElementRemoved

Description: ElementRemoved is used to handle the circustance of elements being removed from an unordered collection. For example, if an element is removed from a bag object, then the iterator must be informed. For this purpose whenever an element is removed from the collection object, this method is called.

NOTE: This method should NEVER be called directly but only by the collection object as a side-effect of removing an element from the collection that has an iterator.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Variant,In)Item:P%Item,
(Optional,Type=Numeric,In)Position:P%Pos

Interface: Primary

Description: Contains all abstract public services for subclass collection iterators.

Method: Last

Description: Last returns the last element of the Set collection based on the current direction of the iterator. If the direction is backwards, the "last" element is the first one.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Next

Description: Next returns the next element in the Set collection. Null is returned if the end of the collection is reached.

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

I%HasNull

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$SortsAfterCriteria

Description: Implements the sorts after criteria operations.

Type: Concrete
Superclasses: RelationalCriteria

Interface: Factory

Description: Interface containing all constructor and destructors.

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the class.

Method: Matches

Description: Returns true if the specified object property sorts after the specified criteria.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)o:P%Object

Instance Variables:

Class: Base$Stream

Description: Provides an abstract, generic IO interface based on the M Open, Use, Read, Write and Close commands. Services are implemented within the specific subclasses.

Type: Concrete

Interface: Factory

Description: Interface the contains constructors and desctructors.

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: This is the public interface for all abstract Stream services.

Method: Commit

Description: Commit is the abstract template for implementing a commit stream function.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Read

Description: Read is the abstract template for implementing a read from stream function.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)Len:P%Len

Method: ReadLine

Description: ReadLine is the abstract template for implementing a read a line stream function.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Reset

Description: Reset is the abstract template for implementing a reset stream function.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Write

Description: Write is the abstract template for implementing a write to stream function.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Text:P%Text

Method: WriteLine

Description: WriteLine is the abstract template for implementing a write line to stream function.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Line:P%Line

Instance Variables:

I%ErrorId

Description: Contains the ID of the error.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%ErrorText

Description: Contains the text of the error.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%ProxyType

Description: Used externally.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1004
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Text

Description: A block of text. Supports infinite text length. When requesting or using text larger than the maximum local string the behavior is undefined. (In the future we expect it to return another text object).

Type: Concrete
Superclasses: EOText

Interface: Factory

Description: Interface the contains constructors and desctructors for the Text class.

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Interface the contains internally used services.

Method: CreateLineMap

Description: Creates a line map for the current Text object. Line map is stored in the LineMap variable.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Dump

Description: Displays the content of the Text object in the Output window. Used for diagnostic work.

Options: Public,Method,Void,Platform=All
Parameters: None

Interface: Primary

Description: This is the public interface for the Text services.

Method: Append

Description: Append a new string onto an existing string.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)String:P%InString

Method: AppendLine

Description: Append a new text line onto an existing string.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Line:P%Line

Method: AppendTextObject

Description: Appends contents of a new Text object onto an existing Text object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Text,In)P%TextObj

Method: BlockCount

Description: Returns the block count of the current Text object.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Numeric,In)BlockSize:P%BlockSize=$$MAXSTR^VESOOPRI

Method: Blocks

Description: Calculates and rounds the block number that is obtained by dividing the text length by a given block size.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=Numeric,In)BlockSize:P%BlockSize=$$MAXSTR^VESOOPRI

Method: Clear

Description: Clear the contents in the text object.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Copy

Description: Copied the contents of a specified Text object into the current object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Text,In)Source:P%Source

Method: GetBlock

Description: Extracts part of element of text by indexing the block number and block size.

Options: Public,Method,Void,Platform=All
Parameters:

(Required,Type=Numeric,In)BlockNumber:P%BlockNumber,
(Required,Type=Numeric,In)BlockSize:P%BlockSize

Method: GetDimension

Description: Returns the length of the text string.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: GetLine

Description: Returns a specified line from the Text object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)LineNumber:P%LineNumber

Method: GetStream

Description: Create a Text>Stream object and returns its OID.

Options: Public,Method,Void
Parameters: None

Method: GetSubstring

Description: Extract a section of string at given start position of the text. (Currently not implemented.)

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Numeric,In)StartPosition:P%StartPosition,
(Required,Type=Numeric,In)StringSize:P%StringSize

Method: GetText

Description: Get whole text string. (Currently not implemented.)

Options: Public,Method,Void,Platform=All
Parameters: None

Method: HasContent

Description: Returns true if the text object has any nonempty lines.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: IsEmpty

Description: Returns true if the Text object is empty.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: SetDimension

Description: Set the length of the text string. (Currently not implemented)

Options: Public,Method,Void,Platform=All
Parameters: None

Method: SetText

Description: Set text contents for the text object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)TextString:P%TextString

Property: LineCount

Description: Returns the number of lines of Text in the object - Each line is delimited by a cf/lf.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

I%Blocks

Description: Internal count of the number of $Maxstr blocks are present.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%LastLen

Description: The size of the final text block.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%Length

Description: The total number of characters in the text.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%LineMap

Description: An internal mapping of the lines and the associated blocks. This is stored as an array.
I%LineMap(Line)=Start Block^StartCharacter^EndBlock^EndCharacter.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%Lines

Description: The number of lines of text (CR/LF) pieces.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%MapValid

Description: Stores a boolean value that indicates whether the line map stored in the variable LineMap is valid for the existing text object.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%ProxyType

Description: Used Externally.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1003
Creation Options: Manually Maintained=No, Dependent=Yes

I%Text

Description: An array of text blocks. Size = $Maxstr except for the final block which is of size I%LastLen.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=""
Creation Options: Manually Maintained=No, Dependent=Yes

I%Valid

Description: Not used.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Text>Stream

Description: The Stream nested class provides streaming functionality for the Text transport type. It inherits from the Base$Stream abstract class.

Type: Concrete
Superclasses: Stream

Interface: Factory

Description: Interface the contains constructors and desctructors for the Text Stream class.

Method: CREATE

Description: Initializes all Stream instance variables of the internal references of the parent Text object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Text,In)I%Parent

Method: InitAllSysVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Internal

Description: Interface the contains internally used services.

Method: UpdateInfo

Description: Initializes all Stream instance variables to the internal references of the parent Text object.

Options: Public,Method,Void,Platform=All
Parameters: None

Interface: Primary

Description: This is the public interface for the Text Stream services.

Method: Commit

Description: Commit implements the commit function for the text stream.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Read

Description: Reads the specified number of characters from the stream.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Numeric,In)Len:P%Len

Method: ReadLine

Description: Reads the specified number of characters from the stream.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Reset

Description: Resets the Stream to the start (For read).

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Write

Description: Writes data to the Stream.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Text:P%Text

Method: WriteLine

Description: Writes to a Stream apennding a cr/lf.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Line:P%Line

Instance Variables:

I%Blocks

Description: Internal count of the number of $Maxstr blocks are present.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%CurBlock

Description: Identifies the "current" block of ESI$Text from which the stream reader will access data.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1
Creation Options: Manually Maintained=No, Dependent=Yes

I%CurChar

Description: Identifies the "current" character within the current block of ESI$Text from which the stream reader will access data.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1
Creation Options: Manually Maintained=No, Dependent=Yes

I%LastLen

Description: The size of the final text block.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0
Creation Options: Manually Maintained=No, Dependent=Yes

I%Length

Description: The total number of characters in the text.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=0

Creation Options: Manually Maintained=No, Dependent=Yes

I%Parent

Description: Pointer to parent transport object.

Inheritable: Yes
Initialization: Static
Binding: Base$Text
Creation Options: Manually Maintained=No, Dependent=No
Creation Keyword: CHILD=1
Creation Parameters:

I%Root

Description: The global root to use for the stream.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%Valid

Description: Contains the validity truth value.

Inheritable: Yes
Initialization: Dynamic
Binding: Expression=1
Creation Options: Manually Maintained=No, Dependent=Yes

I%VectLength

Description: Contains the length of the global vector.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Time

Description: Time objects contain a particular time in a given twenty four hour period. No date is associated with the time object. The granularity of the time object extends to the number of seconds. Manipulation of times is done by the methods and properties associated with a time object.

Type: Concrete
Superclasses: Immutable

Interface: Factory

Description: Interface that contains all constructor and destructor operations for the Time classes.

Method: CREATE

Description: CREATE creates a time object. No date is associated with the time object. The granularity of the time object extends to the number of seconds.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=String,In)Time:P%Time=$Horolog

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the Time class.

Method: Add

Description: Add performs an aritmetic add of an time or interval object to the current time object and returns a new object with a new time. Note: if the resulting time exceeds 24 hours, the value of the time wraps to the next appropriate 24 period.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=String,In)Interval:P%Interval

Method: FromString

Description: Constructs a Time Object from a String. Returns a Null string if the String is invalid. String formats are:

1) "NOW"
2) Mumps $H second piece (Seconds since midnight)
3) HH:MM:SS (Military) 0-23:0-60-0
4) HH:MM:SS AM or PM

Note: When AM & PM are not specificied 12:10 is 12:10 PM and 0:10 is 12:10 AM.

Options: Public,Method,Void,Static,Platform=All

Parameters: (Optional,Type=String,In)P%Time=$Piece($Horolog,",",2)

Method: GreaterThan

Description: GreaterThan returns a truth value that reflects whether or not the time object that is passed in is greater than the time associated with the current time object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Time,In)Time:P%Time

Method: GreaterThanOrEqual

Description: GreaterThanOrEqual returns a truth value that reflects whether or not the time object that is passed in is greater than or equal to the time associated with the current time object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Time,In)Time:P%Time

Method: IsEqual

Description: IsEqual returns a truth value that reflects whether or not the time object that is passed in is equal to the time associated with the current time object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Time,In)Time:P%Time

Method: IsValid

Description: IsValid returns whether or not the value contained in the time object is a valid time or now. Creation of an invalid date can be done. This can be determined by the IsValid method.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: LessThan

Description: LessThan returns a truth value that reflects whether or not the time object that is passed in is less than the time associated with the current time object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Time,In)Time:P%Time

Method: LessThanOrEqual

Description: LessThanOrEqual returns a truth value that reflects whether or not the time object that is passed in is less than or equal to the time associated with the current time object.

Options: Public,Method,Void,Platform=All

Parameters: (Required,Type=Base$Time,In)Time:P%Time

Method: NotEqual

Description: NotEqual returns a truth value that reflects whether or not the time object that is passed in is unequal to the time associated with the current time object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$Time,In)Time:P%Time

Method: Subtract

Description: Subtract takes the current time object and subtracts the passed in time or interval object from it. The resulting object is either a time or interval object depending on the nature of the input.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=String,In)Interval:P%Interval

Property: Current

Description: Current returns the explicit time contained in the time object in 24 hour format: hh:mm:ss.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Hour

Description: Hour returns the hour of the time contained in the time object. This is always in military time. 2:00PM, for example, would return 14.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Minute

Description: Minute returns the minutes of the time contained in the time object. 2:17PM, for example, would return 17.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All

Parameters: None

Property: Second

Description: Second returns the seconds of the time contained in the time object. 2:17:33PM, for example, would return 33.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Class: Base$TimeRange

Description: A Time Range is an association of a time stamp and an interval. With this association two times and the span between those two times can be determined. The starting time is determined from the Time Stamp. The ending time is the time that is generated from the interval added to time stamp. Any time between the beginning and ending time can be determined to be contained within the time range. Times outside this boundary can be determined to be excluded from the time range.

Type: Concrete
Superclasses: Immutable

Interface: Factory

Description: Interface that contains all constructor and destructor operations for the TimeRange class.

Method: CREATE

Description: CREATE creates the time range object. A time range object consists of a time stamp and an associated interval.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Base$TimeStamp,In)TimeStamp:P%TS,
(Required,Type=Base$Interval,In)Interval:P%Int

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the TimeRange class.

Method: IsWithin

Description: IsWithin determines if the time passed in via the input timestamp object is within the span of time described by the time range object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$TimeStamp,In)TimeStamp:P%TS

Method: Overlap

Description: Overlap checks an input time range and determines if any span of time described by the input time range object overlaps the span of time described by the current time range object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)P%TR

Property: Duration

Description: Duration returns the interval contained in the time range. The duration is independant of the starting time.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: End

Description: End returns the end time described in the time range.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Start

Description: Start returns the start time described in the time range.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Class: Base$TimeStamp

Description: TimeStamps are specific time/date combinations that describe an absolute instant of time. Time stamps can be compared, manipulated and used to connect a specific time to an event or action.

Type: Concrete
Superclasses: Immutable

Interface: Factory

Description: Interface that contains all constructor and destructor operations for the TimeStamp class.

Method: CREATE

Description: CREATE creates the time stamp object. For this, the input can be: 1) another timestamp 2) a date/time combination. If nothing is passed in the current date and time are assumed.

Options: Public,Method,Void,Platform=All
Parameters:
(Optional,Type=String,In)TimeStamp:P%TimeStamp=$H,
(Optional,Type=Base$Date,In)Date:P%Date,
(Optional,Type=Base$Time,In)Time:P%Time

Method: InitClassVars

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Interface: Primary

Description: The public interface for the TimeStamp class.

Method: Add

Description: Add performs an aritmetic add of an interval object to the current timestamp object and returns a new object with a new timestamp.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Base$TimeStamp,In)Interval:P%Interval

Method: Date

Description: Returns a date object that contains the date described by the timestamp.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: FromString

Description: Creates a Time Stamp from an input string.

Options: Public,Method,Void,Static,Platform=All
Parameters: (Optional,Type=String,In)TimeStamp:P%TimeStamp

Method: GreaterThan

Description: GreaterThan returns a truth value that reflects whether or not the timestamp object that is passed in is greater than the timestamp associated with the current timestamp object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$TimeStamp,In)TimeStamp:P%TS

Method: GreaterThanOrEqual

Description: GreaterThanOrEqual returns a truth value that reflects whether or not the timestamp object that is passed in is greater than or equal to the timestamp associated with the current timestamp object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$TimeStamp,In)TimeStamp:P%TS

Method: IsBetween

Description: IsBetween determines if the value contained in the timestamp object is between the two timestamps described in the two objects passed in to the method.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Base$TimeStamp,In)P%Start,
(Required,Type=Base$TimeStamp,In)P%End

Method: IsEqual

Description: IsEqual returns a truth value that reflects whether or not the timestamp object that is passed in is equal to the time associated with the current timestamp object.

Options: Public,Method,Void,Platform=All

Parameters: (Required,Type=Base$TimeStamp,In)TimeStamp:P%TS

Method: LessThan

Description: LessThan returns a truth value that reflects whether or not the timestamp object that is passed in is less than the timestamp associated with the current timestamp object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$TimeStamp,In)TimeStamp:P%TS

Method: LessThanOrEqual

Description: LessThanOrEqual returns a truth value that reflects whether or not the timestamp object that is passed in is less than than or equal to the timestamp associated with the current timestamp object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$TimeStamp,In)TimeStamp:P%TS

Method: NotEqual

Description: NotEqual returns a truth value that reflects whether or not the timestamp object that is passed in is unequal to the time associated with the current timestamp object.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Base$TimeStamp,In)TimeStamp:P%TS

Method: Subtract

Description: Subtract takes the current timestamp object and subtracts the passed in interval object from it. The resulting object is a timestamp object.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=Base$TimeStamp,In)Interval:P%Interval

Method: Time

Description: Current returns the explicit time contained in the timestamp object in 24 hour format: hh:mm:ss.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: COASFormat

Description: COASFormat returns the explicit time and date contained in the timestamp object in the following format (which is compatible with the COAS specification - based on the ISO 8601:1988 format):

YYYY-MM-DDThh:mm:ss.dddTZD

for example 1997-07-16T19:20:30.45+01:00

where:

YYYY = four digit year
MM = two digit month (01=January, etc)
DD = two digit day of month (01 through 31)
T = date/time separator
hh = two digits of hour (00 through 23, no AM or PM)
mm = two digits of minute (00 thrugh 59)
ss = two digits of second (00 through 60 ; 60 indicates a positive leap second)
ddd = one or more digits for decimal fraction of a second (no limit on number of digits)
TZD = time zone designator (Z to indicate UTC, or +hh:mm or -hh:mm from UTC)

Partial timestamp formats are allowed, which indicate "unknown" for items omitted. (absence of a time zone designator indicates local time.)

A "?" character can be used as a wildcard.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: CORBAFormat

Description: CORBAFormat returns the explicit time and date contained in the timestamp object in the following format (which is compatible with the COAS specification - based on the ISO 8601:1988 format):

YYYY-MM-DDThh:mm:ss.dddTZD

for example 1997-07-16T19:20:30.45+01:00

where:

YYYY = four digit year
MM = two digit month (01=January, etc)
DD = two digit day of month (01 through 31)
T = date/time separator
hh = two digits of hour (00 through 23, no AM or PM)
mm = two digits of minute (00 thrugh 59)
ss = two digits of second (00 through 60 ; 60 indicates a positive leap second)
ddd = one or more digits for decimal fraction of a second (no limit on number of digits)

TZD = time zone designator (Z to indicate UTC, or +hh:mm or -hh:mm from UTC)

Partial timestamp formats are allowed, which indicate "unknown" for items omitted. (absence of a time zone designator indicates local time.)

A "?" character can be used as a wildcard.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Current

Description: Current returns the explicit time and date contained in the timestamp object in the following format: YYYY-MM-DD hh:mm:ss (24 hour format for the time.)

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Day

Description: Day returns the numeric day of the month of the date contained in the timestamp object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: FileManFormat

Description: Outputs TimeStamp in File Manager format.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Hour

Description: Hour returns the hour of the time contained in the timestamp object. This is always in military time. 2:00PM-2/28/96, for example, would return 14.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Minute

Description: Minute returns the minutes of the time contained in the timestamp object. 2:17PM-2/28/1996, for example, would return 17.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Month

Description: Month returns the numeric month of the date contained in the timestamp object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Mstamp

Description: Mstamp returns the $Horolog form of the stamp for operational purposes.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: OQLFormat

Description: OQLFormat returns the explicit time and date contained in the timestamp object in the following format YYYY-MM-DD hh:mm:ss.ddd

For example 1997-07-16 19:20:30.45

where:

YYYY = four digit year

MM = two digit month (01=January, etc)
DD = two digit day of month (01 through 31)
hh = two digits of hour (00 through 23, no AM or PM)
mm = two digits of minute (00 thrugh 59)
ss = two digits of second (00 through 60 ; 60 indicates a positive leap second)
ddd = one or more digits for decimal fraction of a second (no limit on number of digits)

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Second

Description: Second returns the seconds of the time contained in the timestamp object. 2:17:33PM-2/28/96, for example, would return 33.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: SortNumber

Description: SortNumber returns a sort number by which the date and time value can be sorted. This consists of a date number (e.g., 56671) and a time number (e.g., 35669). The sort number of these valuse should be: 56671.35669

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Property: Year

Description: Year returns the numeric year of the date contained in the timestamp object.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Class: Base$Tools

Description: An abstract class whose subclasses implement various utility tools.

Type: Abstract

Interface: Factory

Description: The Factory interface contains all constructor and destructor services for the Tools subclasses class.

Method: InitAllSysVars

Description: Initializes all variables that flagged as Initialized.

Options: Public,Method,Void
Parameters: None

Method: InitSysVars

Options: Public,Method,Void
Parameters: None

Instance Variables:

I%FindCriteria

Description: Bound to a ESI$FindCriteria object that contains the settable flags that control the search across the EsiObjects structure. These flags are checked in the AcceptVisitor method that exists on all structure levels.

Inheritable: Yes
Initialization: Dynamic
Binding: ESI$FindCriteria
Creation Options: Manually Maintained=No, Dependent=Yes

I%Visitor

Description: Bound to a ESI$DocumentationVisitor object that contains the operations for the DocBuilder iterator.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$Versioned

Description: Mix in class to support object versioning.

Type: Mix-In

Interface: Version

Description: The interface holds all public services for the Version mixin class.

Method: CheckUpgrade

Description: Checks whether an upgrade is to be run based on the

Options: Public,Method,Void,Inheritable,Platform=All
Parameters: (Required,Type=String,In)P%ReqVer

Method: CompareVersions

Options: Public,Method,Void,Inheritable,Static,Platform=All
Parameters:
(Required,Type=String,In)Version1:P%Ver1,
(Required,Type=String,In)Version2:P%Ver2

Method: Upgrade

Options: Public,Method,Void,Inheritable,Platform=All
Parameters: (Required,Type=String,In)Version:P%Version

Property: Version

Description: The Version of the Object. Version numbers are a set of numbers seperated by periods.

e.g.

1.0.0 or
1.0

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Inheritable,Platform=All
Parameters: None

Instance Variables:

I%Version

Description: Contains the Version number (string in the form of 1, 1.1, 1.1.2, etc.) of the current object.

Inheritable: Yes

Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$XMLContentHandler

Description: Abstract class for defining XML document content handling. Contains the abstract callback methods for the XML parser. The individual application classes subclass to this class. The inherited methods in the primary interface are then filled out handle the specific callback requirements for that application.

Type: Abstract
Superclasses: ContentHandler

Interface: Primary

Description: Contains the fundamental abstract call back services for the XML parser.

Method: characters

Description: Abstract method that handles the characters of an XML entity.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Optional,Type=String,In)String:P%String="",
(Optional,Type=Numeric,In)Start:P%Start=1,
(Optional,Type=Numeric,In)Length:P%Len=1

Method: endDocument

Description: Abstract method that provides specilization for the end of the document parse.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: endElement

Description: Abstract method that provides specilization for the end of the element parse.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=String,In)P%uri,
(Required,Type=String,In)P%localName,
(Required,Type=String,In)P%qName

Method: processingInstruction

Description: Abstract method that pocesses application specific information embedded in the XML document.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=Object,In)Target:P%Target,
(Required,Type=Object,In)Data:P%Data

Method: startDocument

Description: Abstract method that provides specilization for the start of the document parse.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: None

Method: startElement

Description: Abstract method that provides specilization for the start of the element parse.

Options: Public,Method,Void,Platform=All,Abstract
Parameters:
(Required,Type=String,In)P%uri,
(Required,Type=String,In)lName:P%localName,
(Required,Type=String,In)qName:P%qName,
(Required,Type=String,In)Attributes:P%atts

Interface: Utility

T

Method: Normalize

Description: Normalizes character data to XML format.

Warning: This operation should not be appllied to text that has already be normalized.
< becomes <
> becomes >
" becomes "
' becomes '
& becomes &

Control Character (Excluding cf/lf) will be converted to esacped form &#dec;

Options: Public,Method,Void,Static,Platform=All

Parameters: (Optional,Type=String,In)P%Value

Method: ToText

Description: This function will convert Normalize XML text into standard text.

Warning: Characters that cannot be converted will be left in the &# or &#x form.

Options: Public,Method,Void,Static,Platform=All
Parameters: (Required,Type=String,In)Text:P%Text

Class: Base$XMLReader

Description: Abstract class that implements the fundamental services required for an XML document reader.

Type: Abstract
Superclasses: Reader

Interface: Primary

Description: Public interface for the XMLReader.

Method: parse

Description: Placeholder method for main parser entry point. Real parser code is in XMLReaderImpl subclass.

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Object,In)TextStream:P%TextStream

Method: setContentHandler

Desciption: Entry point to associate a ContentHandler object with the parser. contentHandler is passed in. It is an object belonging to a suitable concrete subclass of Base$ContentHandler
(e.g. Base$Test>ContentHandler).

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Base$XMLContentHandler,In)Handler:P%contentHandler

Method: setErrorHandler

Description: Entry point to associate an ErrorHandler object with the parser. Note: Error Handlers are not supported at the present time! The error handler object is passed in. It is an object belonging to a suitable concrete subclass of Base$ErrorHandler
(No such class is presently defined).

Options: Public,Method,Void,Platform=All,Abstract
Parameters: (Required,Type=Object,In)errorHanderl:P%errorHandler

Instance Variables:

I%ContentHandler

Description: Parser's content handler object, belonging to a suitable concrete subclass of Base$ContentHandler
(e.g. Base$Test>ContentHandler). The parser sends its results to an application by invoking callback methods on the ContentHandler object.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%ErrorHandler

Description: ErrorHandler object. Not currently implemented.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class: Base$XMLReaderImpl

Description: This is the parser class. It implements the abstract interface specified by the XMLReader superclass.

Note: Because instance variables may hold long strings up to 32K, this class should never be instantiated into a global.

Type: Concrete
Superclasses: XMLReader

Interface: Internal

Description: The private Internal interface contains methods invoked by the paser for each part of an XML document.

Method: Attribute

Description: Parses a single attribute specification, syntactic element "Attribute".

Options: Public,Method,Void,Platform=All
Parameters: None

Method: CharRef

Description: Parses character references. Only ASCII characters are supported. Appends character to I%CharData. Note: If this method is moved into VESOUXML, its contract

should be changed to return the referenced character instead of appending to I%CharData.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: DoctypeDecl

Description: A DTD (Document Type Declaration) consists of a <!DOCTYPE> element. This is not supported by the Parser, and an error occurs if one is detected.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: EncodingDecl

Description: Parses the encoding declaration within an XML declaration.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Boolean,In)Required:P%Reqd

Method: EntityRef

Description: Parses entity references in text. Only the predefined entity references < > & " &apos are supported now. Other entity references may be defined in DTDs, which the Parser does not currently handle. Note: If this method is moved into VESOUXML, its contract should be changed to return the entity contents instead of appending to I%CharData.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Misc

Description: Parses the syntactic element Misc*, which contains Comments, PIs (Processor Instructions) and Whitespace. All of these elements are handled by calls to entry points in the VESOUXML routine.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: SDDecl

Description: Parses the standalone document declaration within an XML declaration. Note: Parser discards the value, even though DTDs are not supported and <!DOCTYPE> elements cannot be parsed.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: SpecialAttribute

Description: Parses the special attributes in XML declarations.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=String,In)Name:P%Nm,
(Required,Type=Boolean,In)Required:P%Reqd

Method: VersionInfo

Description: Parses the version number within an XML declaration.

Options: Public,Method,Void,Platform=All
Parameters:
(Required,Type=Boolean,In)Required:P%Reqd,
(Required,Type=String,In)Version:P%Version

Method: XMLDecl

Description: Entry point to parse the XML Declaration, syntactic element "XMLDecl", that is found at the beginning of XML documents.

Format: <?xml version="1.0" encoding="..." standalone="yes" ?>

Options: Public,Method,Void,Platform=All
Parameters: None

Method: content

Description: Parses text between element start and end tags, syntactic element "content". Comments, CDATA sections, and PIs are handled via calls into VESOUXML. Invokes the characters callback in the ContentHandler.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: document

Description: Entry point to parse the top-level syntactic element "document".

Options: Public,Method,Void,Platform=All
Parameters: None

Method: element

Description: This method parses XML elements. An XML document contains one top-level element, which may contain additional hierarchically nested elements. Namespaces are currently supported for element names. This method invokes the startElement and endElement callbacks in the ContentHandler.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: main

Description: This method is invoked by PARSE^VESOUXML via the callable interface (VESOEX). It handles top-level parsing of a document. It invokes the startDocument and endDocument callbacks on the ContentHandler.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: prolog

Description: Entry point to parse the syntactic element "prolog", which precedes the top-level XML element in a document.

Options: Public,Method,Void,Platform=All
Parameters: None

Interface: Primary

Description: Contains the public services for the XML parser.

Method: ShowError

Description: Instead of supporting the ErrorHandler interface, the Parser provides a ShowError method to display the error text and line number on which an error was detected.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: getError

Description: Returns the error string.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: parse

Description: Main entry point for parsing an XML document. Parser invokes the Read method on the input stream object. Parser should be able to accept as input objects of the following classes:

Base$TextStream (a streaming subclass of ESI$Text)
ESI$ServerFileStream.

Options: Public,Method,Void,Platform=All
Parameters: (Required,Type=Object,In)TextStream:P%TextStream=""

Instance Variables:

I%Attrs

Description: Attribute information for the parser. An object of class Base$AttributesImpl.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%CharData

Description: String variable that accumulates character data associated with XML element currently being parsed. Because the Parser flushes character data for an element before parsing nested elements, it doesn't need to stack I%CharData information.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%Entities

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%ErrLine

Description: Line number where error was detected.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%Error

Description: Error text, as a string.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%NamespaceMgr

Description: Internal object of class Base$XMLReaderImpl>NamespaceMgr. The parser invokes methods on this object when Namespaces are defined or looked up.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%Namespaces

Description: Boolean flag denoting management of namespaces.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%Stream

Description: Stream object containing XML document being parsed. It is passed into Parser's parse method by the application that invokes the Parser.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

Class:

Base$XMLReaderImpl>NamespaceMgr

Description: Nested class that manages Namespace definitions.

Type: Concrete

Interface: Primary

Description: Public interface for the Namespace Manager class.

Method: Define

Description: Defines a Namespace at the current level. Stores the definition into either I%Default or I%URI.

Options: Public,Method,Void,Platform=All
Parameters:

(Optional,Type=String,In)Name:P%Name,
(Optional,Type=String,In)Value:P%Value

Method: Lookup

Description: Looks up a Namespace name and returns the corresponding URI. Finds the definition at the highest (innermost) level.

Options: Public,Method,Void,Platform=All
Parameters: (Optional,Type=String,In)Name:P%Name

Method: LookupDefault

Description: Look up and return default Namespace name applying to the current level.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Pop

Description: Invoked when leaving an element level.

Options: Public,Method,Void,Platform=All
Parameters: None

Method: Push

Description: Invoked when entering an element level.

Options: Public,Method,Void,Platform=All
Parameters: None

Property: LocalDefine

Description: Returns a value that is nonzero if at least one Namespace has been defined at the current level.

Options: Public,Inheritable

Accessor: Value

Options: Public,Method,Void,Platform=All
Parameters: None

Instance Variables:

I%Default

Description: Holds the currently defined Default namespace. Represented as an array, indexed by level, which may assume values between 1 and I%Level.

Inheritable: Yes
Initialization: Static

Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%Level

Description: Contains the current hierarchical naming level, starting from 1.

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes

I%URI

Description: Array representing the mapping of Namespace names to URIs. Indexed by level:

I%URI(level,name)=URI

Inheritable: Yes
Initialization: Static
Binding: Expression
Creation Options: Manually Maintained=No, Dependent=Yes