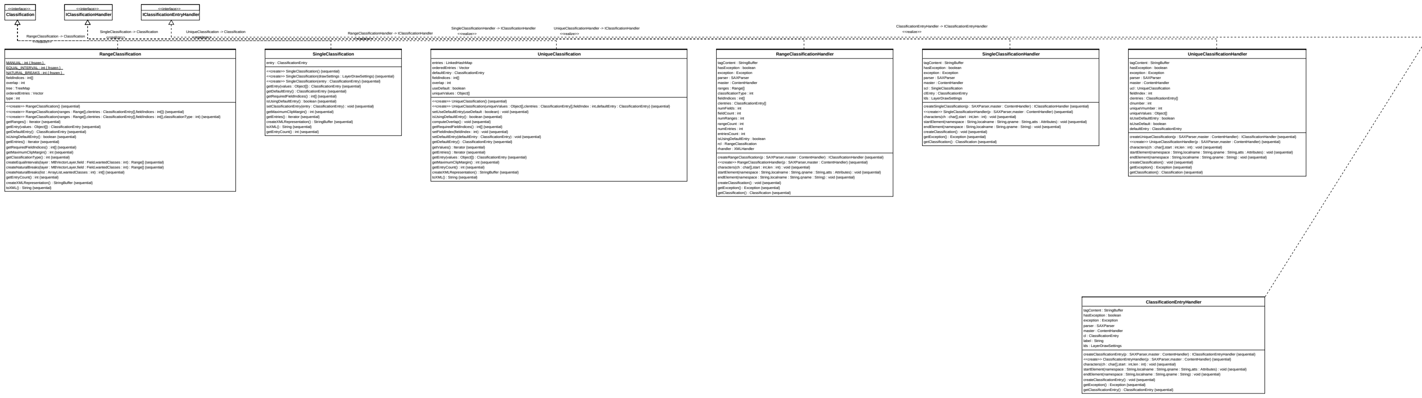
<<interface>>
Classification
<<interface>>
IClassificationHandler
<<interface>>
IClassificationEntryHandler
RangeClassification -> Classification
<<realize>>
SingleClassification -> Classification
<<realize>>
UniqueClassification -> Classification
<<realize>>
RangeClassificationHandler -> IClassificationHandler
<<realize>>
SingleClassificationHandler -> IClassificationHandler
<<realize>>
UniqueClassificationHandler -> IClassificationHandler
<<realize>>
ClassificationEntryHandler -> IClassificationEntryHandler
<<realize>>
RangeClassification
MANUAL : int { frozen }
EQUAL_INTERVAL : int { frozen }
NATURAL_BREAKS : int { frozen }
fieldIndices : int[]
overlap : int
tree : TreeMap
orderedEntries : Vector
type : int
<<create>> RangeClassification() {sequential}
<<create>> RangeClassification(ranges : Range[],clentries : ClassificationEntry[],fieldIndices : int[]) {sequential}
<<create>> RangeClassification(ranges : Range[],clentries : ClassificationEntry[],fieldIndices : int[],classificationType : int) {sequential}
getRanges() : Iterator {sequential}
getEntry(values : Object[]) : ClassificationEntry {sequential}
getDefaultEntry() : ClassificationEntry {sequential}
isUsingDefaultEntry() : boolean {sequential}
getEntries() : Iterator {sequential}
getRequiredFieldIndices() : int[] {sequential}
getMaximumClipMargin() : int {sequential}
getClassificationType() : int {sequential}
createEqualIntervals(layer : MBVectorLayer,field : Field,wantedClasses : int) : Range[] {sequential}
createNaturalBreaks(layer : MBVectorLayer,field : Field,wantedClasses : int) : Range[] {sequential}
createNaturalBreaks(list : ArrayList,wantedClasses : int) : int[] {sequential}
getEntryCount() : int {sequential}
createXMLRepresentation() : StringBuffer {sequential}
toXML() : String {sequential}
SingleClassification
entry : ClassificationEntry
<<create>> SingleClassification() {sequential}
<<create>> SingleClassification(drawSettings : LayerDrawSettings) {sequential}
<<create>> SingleClassification(entry : ClassificationEntry) {sequential}
getEntry(values : Object[]) : ClassificationEntry {sequential}
getDefaultEntry() : ClassificationEntry {sequential}
getRequiredFieldIndices() : int[] {sequential}
isUsingDefaultEntry() : boolean {sequential}
setClassificationEntry(entry : ClassificationEntry) : void {sequential}
getMaximumClipMargin() : int {sequential}
getEntries() : Iterator {sequential}
createXMLRepresentation() : StringBuffer {sequential}
toXML() : String {sequential}
getEntryCount() : int {sequential}
UniqueClassification
entries : LinkedHashMap
orderedEntries : Vector
defaultEntry : ClassificationEntry
fieldIndices : int[]
overlap : int
useDefault : boolean
uniqueValues : Object[]
<<create>> UniqueClassification() {sequential}
<<create>> UniqueClassification(uniqueValues : Object[],clentries : ClassificationEntry[],fieldIndex : int,defaultEntry : ClassificationEntry) {sequential}
setUseDefaultEntry(useDefault : boolean) : void {sequential}
isUsingDefaultEntry() : boolean {sequential}
computeOverlap() : void {sequential}
getRequiredFieldIndices() : int[] {sequential}
setFieldIndex(fieldIndex : int) : void {sequential}
setDefaultEntry(defaultEntry : ClassificationEntry) : void {sequential}
getDefaultEntry() : ClassificationEntry {sequential}
getValues() : Iterator {sequential}
getEntries() : Iterator {sequential}
getEntry(values : Object[]) : ClassificationEntry {sequential}
getMaximumClipMargin() : int {sequential}
getEntryCount() : int {sequential}
createXMLRepresentation() : StringBuffer {sequential}
toXML() : String {sequential}
RangeClassificationHandler
tagContent : StringBuffer
hasException : boolean
exception : Exception
parser : SAXParser
master : ContentHandler
ranges : Range[]
classificationType : int
fieldIndices : int[]
clentries : ClassificationEntry[]
numFields : int
fieldCount : int
numRanges : int
rangeCount : int
numEntries : int
entriesCount : int
isUsingDefaultEntry : boolean
rcl : RangeClassification
rhandler : XMLHandler
createRangeClassification(p : SAXParser,master : ContentHandler) : IClassificationHandler {sequential}
<<create>> RangeClassificationHandler(p : SAXParser,master : ContentHandler) {sequential}
characters(ch : char[],start : int,len : int) : void {sequential}
startElement(namespace : String,localname : String,qname : String,atts : Attributes) : void {sequential}
endElement(namespace : String,localname : String,qname : String) : void {sequential}
createClassification() : void {sequential}
getException() : Exception {sequential}
getClassification() : Classification {sequential}
SingleClassificationHandler
tagContent : StringBuffer
hasException : boolean
exception : Exception
parser : SAXParser
master : ContentHandler
scl : SingleClassification
clEntry : ClassificationEntry
lds : LayerDrawSettings
createSingleClassification(p : SAXParser,master : ContentHandler) : IClassificationHandler {sequential}
<<create>> SingleClassificationHandler(p : SAXParser,master : ContentHandler) {sequential}
characters(ch : char[],start : int,len : int) : void {sequential}
startElement(namespace : String,localname : String,qname : String,atts : Attributes) : void {sequential}
endElement(namespace : String,localname : String,qname : String) : void {sequential}
createClassification() : void {sequential}
getException() : Exception {sequential}
getClassification() : Classification {sequential}
UniqueClassificationHandler
tagContent : StringBuffer
hasException : boolean
exception : Exception
parser : SAXParser
master : ContentHandler
ucl : UniqueClassification
fieldIndex : int
clentries : ClassificationEntry[]
clnumber : int
uniqueVnumber : int
uniqueValues : Object[]
isUseDefaultEntry : boolean
isUseDefault : boolean
defaultEntry : ClassificationEntry
createUniqueClassification(p : SAXParser,master : ContentHandler) : IClassificationHandler {sequential}
<<create>> UniqueClassificationHandler(p : SAXParser,master : ContentHandler) {sequential}
characters(ch : char[],start : int,len : int) : void {sequential}
startElement(namespace : String,localname : String,qname : String,atts : Attributes) : void {sequential}
endElement(namespace : String,localname : String,qname : String) : void {sequential}
createClassification() : void {sequential}
getException() : Exception {sequential}
getClassification() : Classification {sequential}
ClassificationEntryHandler
tagContent : StringBuffer
hasException : boolean
exception : Exception
parser : SAXParser
master : ContentHandler
cl : ClassificationEntry
label : String
lds : LayerDrawSettings
createClassificationEntry(p : SAXParser,master : ContentHandler) : IClassificationEntryHandler {sequential}
<<create>> ClassificationEntryHandler(p : SAXParser,master : ContentHandler) {sequential}
characters(ch : char[],start : int,len : int) : void {sequential}
startElement(namespace : String,localname : String,qname : String,atts : Attributes) : void {sequential}
endElement(namespace : String,localname : String,qname : String) : void {sequential}
createClassificationEntry() : void {sequential}
getException() : Exception {sequential}
getClassificationEntry() : ClassificationEntry {sequential}