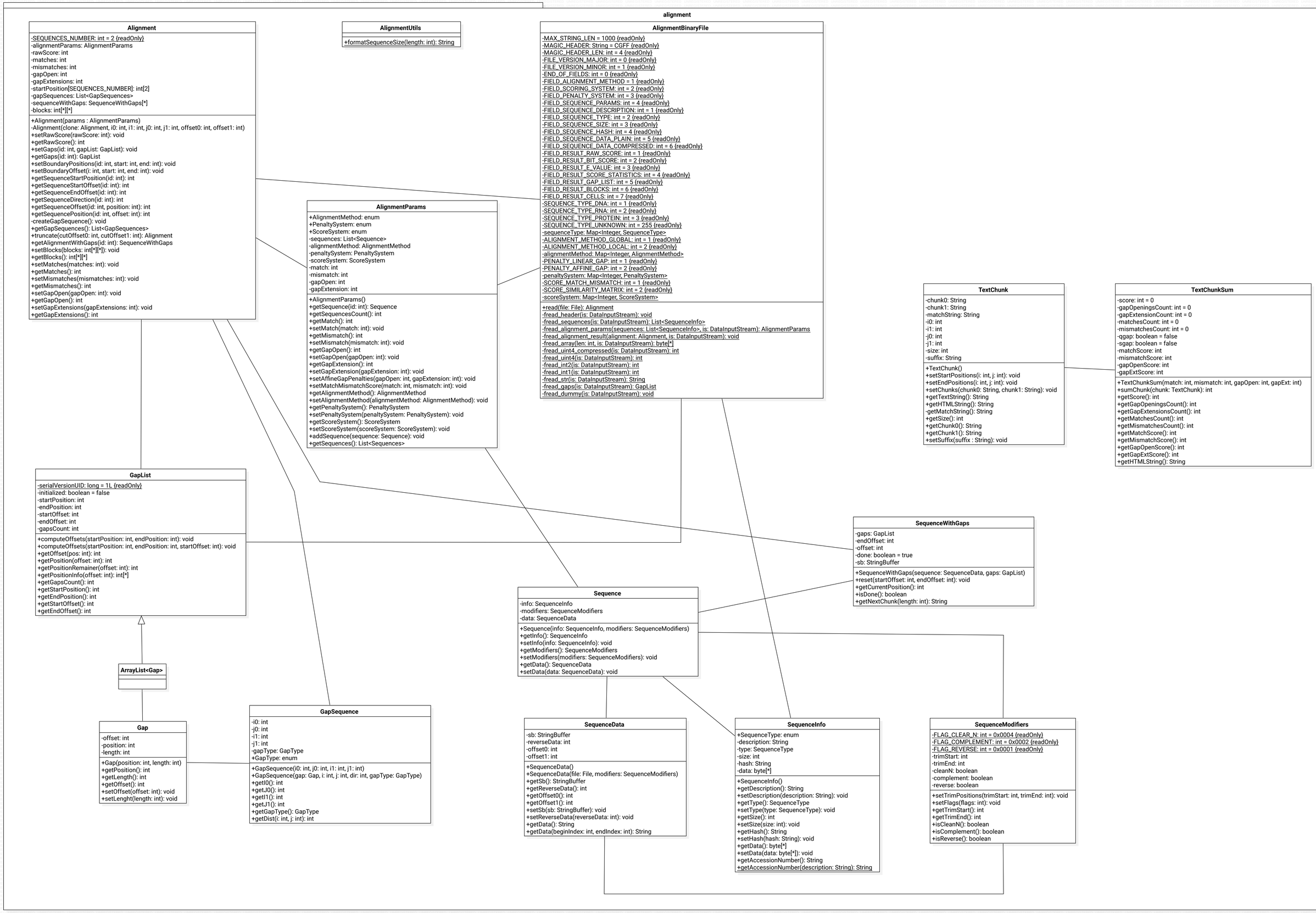
alignment
Alignment
-SEQUENCES_NUMBER: int = 2 {readOnly}
-alignmentParams: AlignmentParams
-rawScore: int
-matches: int
-mismatches: int
-gapOpen: int
-gapExtensions: int
-startPosition[SEQUENCES_NUMBER]: int[2]
-gapSequences: List<GapSequences>
-sequenceWithGaps: SequenceWithGaps[*]
-blocks: int[*][*]
+Alignment(params : AlignmentParams)
-Alignment(clone: Alignment, i0: int, i1: int, j0: int, j1: int, offset0: int, offset1: int)
+setRawScore(rawScore: int): void
+getRawScore(): int
+setGaps(id: int, gapList: GapList): void
+getGaps(id: int): GapList
+setBoundaryPositions(id: int, start: int, end: int): void
+setBoundaryOffset(i: int, start: int, end: int): void
+getSequenceStartPosition(id: int): int
+getSequenceStartOffset(id: int): int
+getSequenceEndOffset(id: int): int
+getSequenceDirection(id: int): int
+getSequenceOffset(id: int, position: int): int
+getSequencePosition(id: int, offset: int): int
-createGapSequence(): void
+getGapSequences(): List<GapSequences>
+truncate(cutOffset0: int, cutOffset1: int): Alignment
+getAlignmentWithGaps(id: int): SequenceWithGaps
+setBlocks(blocks: int[*][*]): void
+getBlocks(): int[*][*]
+setMatches(matches: int): void
+getMatches(): int
+setMismatches(mismatches: int): void
+getMismatches(): int
+setGapOpen(gapOpen: int): void
+getGapOpen(): int
+setGapExtensions(gapExtensions: int): void
+getGapExtensions(): int
AlignmentUtils
+formatSequenceSize(length: int): String
AlignmentBinaryFile
-MAX_STRING_LEN = 1000 {readOnly}
-MAGIC_HEADER: String = CGFF {readOnly}
-MAGIC_HEADER_LEN: int = 4 {readOnly}
-FILE_VERSION_MAJOR: int = 0 {readOnly}
-FILE_VERSION_MINOR: int = 1 {readOnly}
-END_OF_FIELDS: int = 0 {readOnly}
-FIELD_ALIGNMENT_METHOD = 1 {readOnly}
-FIELD_SCORING_SYSTEM: int = 2 {readOnly}
-FIELD_PENALTY_SYSTEM: int = 3 {readOnly}
-FIELD_SEQUENCE_PARAMS: int = 4 {readOnly}
-FIELD_SEQUENCE_DESCRIPTION: int = 1 {readOnly}
-FIELD_SEQUENCE_TYPE: int = 2 {readOnly}
-FIELD_SEQUENCE_SIZE: int = 3 {readOnly}
-FIELD_SEQUENCE_HASH: int = 4 {readOnly}
-FIELD_SEQUENCE_DATA_PLAIN: int = 5 {readOnly}
-FIELD_SEQUENCE_DATA_COMPRESSED: int = 6 {readOnly}
-FIELD_RESULT_RAW_SCORE: int = 1 {readOnly}
-FIELD_RESULT_BIT_SCORE: int = 2 {readOnly}
-FIELD_RESULT_E_VALUE: int = 3 {readOnly}
-FIELD_RESULT_SCORE_STATISTICS: int = 4 {readOnly}
-FIELD_RESULT_GAP_LIST: int = 5 {readOnly}
-FIELD_RESULT_BLOCKS: int = 6 {readOnly}
-FIELD_RESULT_CELLS: int = 7 {readOnly}
-SEQUENCE_TYPE_DNA: int = 1 {readOnly}
-SEQUENCE_TYPE_RNA: int = 2 {readOnly}
-SEQUENCE_TYPE_PROTEIN: int = 3 {readOnly}
-SEQUENCE_TYPE_UNKNOWN: int = 255 {readOnly}
-sequenceType: Map<Integer, SequenceType>
-ALIGNMENT_METHOD_GLOBAL: int = 1 {readOnly}
-ALIGNMENT_METHOD_LOCAL: int = 2 {readOnly}
-alignmentMethod: Map<Integer, AlignmentMethod>
-PENALTY_LINEAR_GAP: int = 1 {readOnly}
-PENALTY_AFFINE_GAP: int = 2 {readOnly}
-penaltySystem: Map<Integer, PenaltySystem>
-SCORE_MATCH_MISMATCH: int = 1 {readOnly}
-SCORE_SIMILARITY_MATRIX: int = 2 {readOnly}
-scoreSystem: Map<Integer, ScoreSystem>
+read(file: File): Alignment
-fread_header(is: DataInputStream): void
-fread_sequences(is: DataInputStream): List<SequenceInfo>
-fread_alignment_params(sequences: List<SequenceInfo>, is: DataInputStream): AlignmentParams
-fread_alignment_result(alignment: Alignment, is: DataInputStream): void
-fread_array(len: int, is: DataInputStream): byte[*]
-fread_uint4_compressed(is: DataInputStream): int
-fread_uint4(is: DataInputStream): int
-fread_int2(is: DataInputStream): int
-fread_int1(is: DataInputStream): int
-fread_str(is: DataInputStream): String
-fread_gaps(is: DataInputStream): GapList
-fread_dummy(is: DataInputStream): void
AlignmentParams
+AlignmentMethod: enum
+PenaltySystem: enum
+ScoreSystem: enum
-sequences: List<Sequence>
-alignmentMethod: AlignmentMethod
-penaltySystem: PenaltySystem
-scoreSystem: ScoreSystem
-match: int
-mismatch: int
-gapOpen: int
-gapExtension: int
+AlignmentParams()
+getSequence(id: int): Sequence
+getSequencesCount(): int
+getMatch(): int
+setMatch(match: int): void
+getMismatch(): int
+setMismatch(mismatch: int): void
+getGapOpen(): int
+setGapOpen(gapOpen: int): void
+getGapExtension(): int
+setGapExtension(gapExtension: int): void
+setAffineGapPenalties(gapOpen: int, gapExtension: int): void
+setMatchMismatchScore(match: int, mismatch: int): void
+getAlignmentMethod(): AlignmentMethod
+setAlignmentMethod(alignmentMethod: AlignmentMethod): void
+getPenaltySystem(): PenaltySystem
+setPenaltySystem(penaltySystem: PenaltySystem): void
+getScoreSystem(): ScoreSystem
+setScoreSystem(scoreSystem: ScoreSystem): void
+addSequence(sequence: Sequence): void
+getSequences(): List<Sequences>
TextChunk
-chunk0: String
-chunk1: String
-matchString: String
-i0: int
-i1: int
-j0: int
-j1: int
-size: int
-suffix: String
+TextChunk()
+setStartPositions(i: int, j: int): void
+setEndPositions(i: int, j: int): void
+setChunks(chunk0: String, chunk1: String): void
+getTextString(): String
+getHTMLString(): String
-getMatchString(): String
+getSize(): int
+getChunk0(): String
+getChunk1(): String
+setSuffix(suffix : String): void
TextChunkSum
-score: int = 0
-gapOpeningsCount: int = 0
-gapExtensionCount: int = 0
-matchesCount: int = 0
-mismatchesCount: int = 0
-qgap: boolean = false
-sgap: boolean = false
-matchScore: int
-mismatchScore: int
-gapOpenScore: int
-gapExtScore: int
+TextChunkSum(match: int, mismatch: int, gapOpen: int, gapExt: int)
+sumChunk(chunk: TextChunk): int
+getScore(): int
+getGapOpeningsCount(): int
+getGapExtensionsCount(): int
+getMatchesCount(): int
+getMismatchesCount(): int
+getMatchScore(): int
+getMismatchScore(): int
+getGapOpenScore(): int
+getGapExtScore(): int
+getHTMLString(): String
GapList
-serialVersionUID: long = 1L {readOnly}
-initialized: boolean = false
-startPosition: int
-endPosition: int
-startOffset: int
-endOffset: int
-gapsCount: int
+computeOffsets(startPosition: int, endPosition: int): void
+computeOffsets(startPosition: int, endPosition: int, startOffset: int): void
+getOffset(pos: int): int
+getPosition(offset: int): int
+getPositionRemainer(offset: int): int
+getPositionInfo(offset: int): int[*]
+getGapsCount(): int
+getStartPosition(): int
+getEndPosition(): int
+getStartOffset(): int
+getEndOffset(): int
SequenceWithGaps
-gaps: GapList
-endOffset: int
-offset: int
-done: boolean = true
-sb: StringBuffer
+SequenceWithGaps(sequence: SequenceData, gaps: GapList)
+reset(startOffset: int, endOffset: int): void
+getCurrentPosition(): int
+isDone(): boolean
+getNextChunk(length: int): String
Sequence
-info: SequenceInfo
-modifiers: SequenceModifiers
-data: SequenceData
+Sequence(info: SequenceInfo, modifiers: SequenceModifiers)
+getInfo(): SequenceInfo
+setInfo(info: SequenceInfo): void
+getModifiers(): SequenceModifiers
+setModifiers(modifiers: SequenceModifiers): void
+getData(): SequenceData
+setData(data: SequenceData): void
ArrayList<Gap>
Gap
-offset: int
-position: int
-length: int
+Gap(position: int, length: int)
+getPosition(): int
+getLength(): int
+getOffset(): int
+setOffset(offset: int): void
+setLenght(length: int): void
GapSequence
-i0: int
-j0: int
-i1: int
-j1: int
-gapType: GapType
+GapType: enum
+GapSequence(i0: int, j0: int, i1: int, j1: int)
+GapSequence(gap: Gap, i: int, j: int, dir: int, gapType: GapType)
+getI0(): int
+getJ0(): int
+getI1(): int
+getJ1(): int
+getGapType(): GapType
+getDist(i: int, j: int): int
SequenceData
-sb: StringBuffer
-reverseData: int
-offset0: int
-offset1: int
+SequenceData()
+SequenceData(file: File, modifiers: SequenceModifiers)
+getSb(): StringBuffer
+getReverseData(): int
+getOffset0(): int
+getOffset1(): int
+setSb(sb: StringBuffer): void
+setReverseData(reverseData: int): void
+getData(): String
+getData(beginIndex: int, endIndex: int): String
SequenceInfo
+SequenceType: enum
-description: String
-type: SequenceType
-size: int
-hash: String
-data: byte[*]
+SequenceInfo()
+getDescription(): String
+setDescription(description: String): void
+getType(): SequenceType
+setType(type: SequenceType): void
+getSize(): int
+setSize(size: int): void
+getHash(): String
+setHash(hash: String): void
+getData(): byte[*]
+setData(data: byte[*]): void
+getAccessionNumber(): String
+getAccessionNumber(description: String): String
SequenceModifiers
-FLAG_CLEAR_N: int = 0x0004 {readOnly}
-FLAG_COMPLEMENT: int = 0x0002 {readOnly}
-FLAG_REVERSE: int = 0x0001 {readOnly}
-trimStart: int
-trimEnd: int
-cleanN: boolean
-complement: boolean
-reverse: boolean
+setTrimPositions(trimStart: int, trimEnd: int): void
+setFlags(flags: int): void
+getTrimStart(): int
+getTrimEnd(): int
+isCleanN(): boolean
+isComplement(): boolean
+isReverse(): boolean