<<class>>
DerivationManager

- derivationAnnotationManager: DerivationAnnotationManager
- fileCopy: FileCopy

+ processDerivation(inputPath, outputPath): void
+ createSoftwareDerivation(inputPath, outputPath, newProjectName, derivationManager): void

<<use>>

<<class>>
ProjectCopier

+ copyExistingProject(pathToProjectTree, pathToNewProject)

<<class>>
DerivationVariableProcessor

- configurationVariableManager: ConfigurationVariableManager

+ andDerivationRecursive(stringJSONObject): boolean
+ andDerivationRecursive(JSONObject): boolean
+ orDerivationRecursive(stringJSONObject): boolean
+ orDerivationRecursive(JSONObject): boolean

<<class>>
DerivationAnnotationManager

- derivationVariableProcessor: DerivationVariableProcessor

+ searchForAnnotation(reader, writer): boolean
- chooseAnnotationMethod(reader, mark, writer, buffer): boolean

<<use>>

<<class>>
ConfigurationVariableManager

- configVariables: Map<String, String>

+ addVariable(variableName, value): void
+ getVariable(variableName): String
+ getAllVariableNames(variableName): String[]

<<class>>
FileCopy

+ processFile(inputPath, outputPath): void

<<class>>
ImportAnnotation

process(reader, writer, buffer): boolean
process(reader, buffer, contentBuffer): boolean
checkAnnotation(stringToCheck): boolean

<<abstract>>
DerivationAnnotation

- classAnnotation: ClassAnnotation
- methodAnnotation: MethodAnnotation
- importAnnotation: ImportAnnotation

 # process(reader, writer, buffer): boolean
 # process(reader, buffer, contentBuffer): boolean
 # checkAnnotation(stringToCheck): boolean
parse(reader, writer, buffer): boolean
parse(reader, buffer, contentBuffer): boolean

<<class>>
MethodAnnotation

process(reader, writer, buffer): boolean
process(reader, buffer, contentBuffer): boolean
checkAnnotation(stringToCheck): boolean

<<class>>
ClassAnnotation

process(reader, writer, buffer): boolean
process(reader, buffer, contentBuffer): boolean
checkAnnotation(stringToCheck): boolean

<<class>>
IncorrectAnnotationUsageException