JavaScript is disabled on your browser.

jminusminus

Class JCompilationUnit

- java.lang.Object
  - jminusminus.JAST
    - jminusminus.JCompilationUnit

.


```
class JCompilationUnit
extends JAST
```

The abstract syntax tree (AST) node representing a compilation unit, and so the root of the AST. It keeps track of the name of the source file, its package name, a list of imported types, a list of type (eg class) declarations, and a flag indicating if a semantic error has been detected in analysis or code generation. It also maintains a CompilationUnitContext (built in pre-analysis) for declaring both imported and declared types. The AST is produced by the Parser. Once the AST has been built, three successive methods are invoked: (1) Method preAnalyze() is invoked for making a first pass at type analysis, recursively reaching down to the member headers for declaring types and member interfaces in the environment (contexts). preAnalyze() also creates a partial class file (in memory) for recording member header information, using the partialCodegen() method. (2) Method analyze() is invoked for type-checking field initializations and method bodies, and determining the types of all expressions. A certain amount of tree surgery is also done here. And stack frame offsets are computed for method parameters and local variables. (3) Method codegen() is invoked for generating code for the compilation unit to a class file. For each type declaration, it instantiates a CLEmmiter object (an abstraction of the class file) and then invokes methods on that CLEmitter for generating instructions. At the end of each type declaration, a method is invoked on the CLEmitter which writes the class out to the file system either as .class file or as a .s (SPIM) file. Of course, codegen() makes recursive calls down the tree, to the codegen() methods at each node, for generating the appropriate instructions.

- **Field Summary**
- **Fields inherited from class jminusminus.JAST**

  `compilationUnit, line`
- **Constructor Summary**

Constructors

Constructor and Description
`JCompilationUnit(String fileName, int line, TypeName packageName, ArrayList<TypeName> imports, ArrayList<JAST> typeDeclarations)` Construct an AST node for a compilation unit given a file name, class directory, line number, package name, list of imports, and type declarations.

- **Method Summary**

Methods

Modifier and Type	Method and Description
`JAST`	**`analyze(Context context)`** Perform semantic analysis on the AST in the specified context.
`ArrayList<CLFile>`	**`clFiles()`** Return the list of CLFile objects corresponding to the type declarations in this compilation unit.
`void`	**`codegen(CLEmitter output)`** Generating code for a compilation unit means generating code for each of the type declarations.
`boolean`	**`errorHasOccurred()`** Has a semantic error occurred up to now?
`String`	**`packageName()`** The package in which this compilation unit is defined.
`void`	**`preAnalyze()`** Construct a context for the compilation unit, initializing it with imported types.
`void`	**`reportSemanticError(int line, String message, Object... arguments)`** Report a semantic error.
`void`	**`writeToStdOut(PrettyPrinter p)`** Write the information pertaining to this AST to STDOUT.

- **Methods inherited from class jminusminus.JAST**

  `line, partialCodegen`
- **Methods inherited from class java.lang.Object**

  `clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

- ## Constructor Detail

  - ### JCompilationUnit

```
publicJCompilationUnit(StringfileName,
                       intline,
                       TypeNamepackageName,
                       ArrayList<TypeName>imports,
                       ArrayList<JAST>typeDeclarations)
```

Construct an AST node for a compilation unit given a file name, class directory, line number, package name, list of imports, and type declarations.

Parameters:

`fileName` - the name of the source file.
`line` - line in which the compilation unit occurs in the source file.
`packageName` - package name.
`imports` - a list of imports.
`typeDeclarations` - type declarations.

- ## Method Detail

  - ### packageName

```
publicStringpackageName()
```

The package in which this compilation unit is defined

Returns:

the package name.

  - ### errorHasOccurred

```
publicbooleanerrorHasOccurred()
```

Has a semantic error occurred up to now?

Returns:

true or false.

  - ### reportSemanticError

```
publicvoidreportSemanticError(intline,
                              Stringmessage,
                              Object...arguments)
```

Report a semantic error.

Parameters:

`line` - line in which the error occurred in the source file.
`message` - message identifying the error.
`arguments` - related values.

  - ### preAnalyze

```
publicvoidpreAnalyze()
```

Construct a context for the compilation unit, initializing it with imported types. Then pre-analyze the unit's type declarations, adding their types to the context.

  - ### analyze

```
publicJASTanalyze(Contextcontext)
```

Perform semantic analysis on the AST in the specified context.

Specified by:
`analyze` in class `JAST`

Parameters:
`context` - context in which names are resolved (ignored here).

Returns:
the analyzed (and possibly rewritten) AST subtree.

- **codegen**

```
publicvoidcodegen(CLEmitteroutput)
```

Generating code for a compilation unit means generating code for each of the type declarations.

Specified by:
`codegen` in class `JAST`

Parameters:
`output` - the code emitter (basically an abstraction for producing the .class file).

- **clFiles**

```
publicArrayList<CLFile>clFiles()
```

Return the list of CLFile objects corresponding to the type declarations in this compilation unit.

Returns:
list of CLFile objects

- **writeToStdOut**

```
publicvoidwriteToStdOut(PrettyPrinterp)
```


Description copied from class: JAST

Write the information pertaining to this AST to STDOUT.

Specified by:
`writeToStdOut` in class `JAST`

Parameters:
p - for pretty printing with indentation.