jminusminus

Class Context

- java.lang.Object
  - jminusminus.Context

- Direct Known Subclasses:
  ClassContext, CompilationUnitContext, LocalContext

```
class Context
extends Object
```

A Context encapsulates the environment in which an AST is analyzed. It represents a scope; the scope of a variable is captured by its context. It's the symbol table. Because scopes are lexically nested in Java (and so in j--), the environment can be seen as a stack of contexts, each of which is a mapping from names to their definitions (IDefns). A Context keeps track of it's (most closely) surrounding context, its surrounding class context, and its surrounding compilation unit context, as well as a map of from names to definitions in the level of scope the Context represents. Contexts are created for the compilation unit (a CompilationUnitContext), a class (a ClassContext), each method (a MethodContext), and each block (a LocalContext). If we were to add the for-statement to j--, we would necessarily create a (local) context. From the outside, the structure looks like a tree strung over the AST. But from any location on the AST, that is from any point along a particular branch, it looks like a stack of context objects leading back to the root of the AST, that is, back to the JCompilationUnit object at the root. Part of this structure is built during pre-analysis; pre-analysis reaches only into the type (eg class) declaration for typing the members; pre-analysis does not reach into the method bodies. The rest of it is built during analysis.

- **Field Summary**

Fields

Modifier and Type	Field and Description

protected ClassContext	**classContext** The surrounding class context.
protected CompilationUnitContext	**compilationUnitContext** The compilation unit context (for the whole source program or file).
protected Map<String,IDefn>	**entries** Map of (local variable, formal parameters, type) names to their definitions.
protected Context	**surroundingContext** The surrounding context (scope).

- **Constructor Summary**

Constructors

Modifier	Constructor and Description
protected	**Context**(Context surrounding, ClassContext classContext, CompilationUnitContext compilationUnitContext) Construct a Context.

- **Method Summary**

Methods

Modifier and Type	Method and Description
void	**addEntry**(int line, String name, IDefn definition) Add an entry to the symbol table, binding a name to its definition in the current context.
void	**addType**(int line, Type type) Add the type to the environment.
ClassContext	**classContext**() Return the surrounding class context.
CompilationUnitContext	**compilationUnitContext**() Return the surrounding compilation unit context.
Type	**definingType**() The type that defines this context (used principally for checking acessibility).
IDefn	**lookup**(String name) Return the definition for a name in the environment.
Type	**lookupType**(String name) Return the definition for a type name in the environment.
MethodContext	**methodContext**() Return the closest surrounding method context.
Set<String>	**names**() The names declared in this context.

Context	**surroundingContext**() Return the surrounding context (scope) in the stack of contexts.
void	**writeToStdOut**(PrettyPrinter p) Write the contents of this context to STDOUT.

- **Methods inherited from class java.lang.Object**

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

- **Field Detail**

  - **surroundingContext**

```
protectedContext surroundingContext
```

The surrounding context (scope).

  - **classContext**

```
protectedClassContext classContext
```

The surrounding class context.

  - **compilationUnitContext**

```
protectedCompilationUnitContext compilationUnitContext
```

The compilation unit context (for the whole source program or file).

  - **entries**

```
protectedMap<String,IDefn> entries
```


Map of (local variable, formal parameters, type) names to their definitions.

- **Constructor Detail**

  - **Context**

```
protectedContext(Contextsurrounding,
       ClassContextclassContext,
       CompilationUnitContextcompilationUnitContext)
```

Construct a Context.

Parameters:

`surrounding` - the surrounding context (scope).
`classContext` - the surrounding class context.
`compilationUnitContext` - the compilation unit context (for the whole source program or file).

- **Method Detail**

  - **addEntry**

```
publicvoidaddEntry(intline,
           Stringname,
           IDefndefinition)
```

Add an entry to the symbol table, binding a name to its definition in the

current context.

Parameters:

`name` - the name being declared.
`definition` - and its definition.

- **lookup**

```
publicIDefnlookup(Stringname)
```

Return the definition for a name in the environment. If it's not found in this context, we look for it in the surrounding context(s).

Parameters:

`name` - the name whose definition we're looking for.

Returns:

the definition (or null, if not found).

- **lookupType**

```
publicTypelookupType(Stringname)
```

Return the definition for a type name in the environment. For now, we look for types only in the CompilationUnitContext.

Parameters:

`name` - the name of the type whose definition we're looking for.

Returns:

the definition (or null, if not found).

- **addType**

```
publicvoidaddType(intline,
                  Typetype)
```


Add the type to the environment.

Parameters:

`line` - the number of type declaration.
`type` - the type we are declaring.

- **definingType**

```
publicTypedefiningType()
```

The type that defines this context (used principally for checking acessibility).

Returns:

the type that defines this context.

- **surroundingContext**

```
publicContextsurroundingContext()
```

Return the surrounding context (scope) in the stack of contexts.

Returns:

the surrounding context.

- **classContext**

```
publicClassContextclassContext()
```

Return the surrounding class context.

Returns:

the surrounding class context.

- **compilationUnitContext**

```
publicCompilationUnitContextcompilationUnitContext()
```

Return the surrounding compilation unit context. This is where imported types and other types defined in the compilation unit are declared.

Returns:

the compilation unit context.

- **methodContext**

```
publicMethodContextmethodContext()
```

Return the closest surrounding method context. Return null if we're not within a method.

Returns:

the method context.

- **names**

```
publicSet<String>names()
```

The names declared in this context.

Returns:

the set of declared names.

- **writeToStdOut**

```
publicvoidwriteToStdOut(PrettyPrinterp)
```

Write the contents of this context to STDOUT.

Parameters:

p - for pretty printing with indentation.