JavaScript is disabled on your browser.

- **Prev Class**
- **Next Class**

- Frames
- No Frames

- All Classes

- Summary:
- Nested |
- Field |
- Constr |
- Method

- Detail:
- Field |
- Constr |
- Method

jminusminus

Class JExpression

- java.lang.Object
  - jminusminus.JAST
    - jminusminus.JStatement
      - jminusminus.JExpression

- Direct Known Subclasses:

  JArrayExpression, JArrayInitializer, JBinaryExpression, JCastOp, JFieldSelection, JInstanceOfOp, JLiteralChar, JLiteralFalse, JLiteralInt, JLiteralNull, JLiteralString, JLiteralTrue, JMessageExpression, JNewArrayOp, JNewOp, JSuper, JSuperConstruction, JThis, JThisConstruction, JUnaryExpression, JVariable, JWildExpression

```
abstract class JExpression
extends JStatement
```

The AST node for an expression. The syntax says all expressions are statements, but a semantic check throws some (those without a side-effect) out. Every expression has a type and a flag saying whether or not it's a statement-expression.

- ### Field Summary

Fields

Modifier and Type	Field and Description
`protected boolean`	**`isStatementExpression`** Whether or not this expression is a statement.
`protected Type`	**`type`** Expression type.

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

  `compilationUnit`, `line`

- **Constructor Summary**

Constructors

Modifier	Constructor and Description
`protected`	`JExpression(int line)` Construct an AST node for an expression given its line number.

- **Method Summary**

Methods

Modifier and Type	Method and Description
`abstract JExpression`	`analyze(Context context)` The analysis of any JExpression returns a JExpression.
`void`	`codegen(CLEmitter output, String targetLabel, boolean onTrue)` Perform (short-circuit) code generation for a boolean expression, given the code emitter, a target label, and whether we branch to that label on true or on false.
`boolean`	`isStatementExpression()` Is this a statementRxpression?
`Type`	`type()` Return the expression type.

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

  `codegen`, `line`, `partialCodegen`, `writeToStdOut`

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

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

- **Field Detail**

  - **type**

  `protectedType type`

  Expression type.

  - **isStatementExpression**

  `protectedboolean isStatementExpression`

  Whether or not this expression is a statement.

Constructor Detail

JExpression

```
protectedJExpression(intline)
```

Construct an AST node for an expression given its line number.

Parameters:

`line` - line in which the expression occurs in the source file.

Method Detail

type

```
publicTypetype()
```

Return the expression type.

Returns:

the expression type.

isStatementExpression

```
publicbooleanisStatementExpression()
```

Is this a statementRxpression?

Returns:

whether or not this is being used as a statement.

analyze

```
public abstractJExpressionanalyze(Contextcontext)
```

The analysis of any JExpression returns a JExpression. That's all this (re-)declaration of analyze() says.

Specified by:

`analyze` in class `JAST`

Parameters:

`context` - context in which names are resolved.

Returns:

the analyzed (and possibly rewritten) AST subtree.

codegen

```
publicvoidcodegen(CLEmitteroutput,
                  StringtargetLabel,
                  booleanonTrue)
```

Perform (short-circuit) code generation for a boolean expression, given the code emitter, a target label, and whether we branch to that label on true or on false.

Parameters:

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

`targetLabel` - the label to which we should branch.

`onTrue` - do we branch on true?

- **Prev Class**
- **Next Class**

- Frames
- No Frames

- All Classes

- Summary:
- Nested |
- Field |
- Constr |
- Method

- Detail:
- Field |
- Constr |
- Method