JavaScript is disabled on your browser.

- Prev Package
- Next Package

- Frames
- No Frames

- All Classes

Package jminusminus

This package contains the scanner, parser, AST node representations, bytecode emitter, JavaCC description file, and driver classes for the j-- compiler.

See: Description

Interface Summary

Interface	Description
Converter	A Converter encapusates any (possibly none) code necessary to perform a cast operation.
IDefn	The IDefn type is used to implement definitions of those things (local variables, formal arguments, types) that are named in some context (or scope).
JavaCCParserConstants	
JLhs	The type of any expression that can appear on the lhs of an assignment statement, i.e., JVariable, JFieldSelection, JArrayExpression.
JMember	An interface supported by all class (or later, interface) members.
JTypeDecl	An interface supported by class (or later, interface) declarations.

Class Summary

Class	Description
AmbiguousName	Ambiguous names are meant to deal with snippets like
ArrayTypeName	The (temporary) representation of an array's type.
Boxing	Boxing requires invoking the appropriate conversion method from the (Java) API.
ByteClassLoader	A class loader to be able to load a class from a byte stream.
CharReader	A buffered character reader.
CLAbsorber	CLAbsorber is for reading a Java class into an in-memory CLFile

	representation and printing it out to STDOUT in a format similar to that of javap.
CLAnnotation	Representation of annotation structure (JVM Spec Section 4.8.15).
CLAnnotationDefaultAttribute	Representation of AnnotationDefault_attribute structure (JVM Spec Section 4.8.2).
CLArithmeticInstruction	Representation for ARITHMETIC1 and ARITHMETIC2 instructions.
CLArrayInstruction	Representation for ARRAY1, ARRAY2 and ARRAY3 instructions.
ClassContext	Represents the context (scope, environment, symbol table) for a type, eg a class, in j--.
CLAttributeInfo	Representation of attribute_info structure (JVM Spec Section 4.8).
CLBitInstruction	Representation for BIT instructions.
CLBranchStack	This class is used for control flow analysis to compute maximum depth of operand stack for a method.
CLBranchTarget	Instances of this class form the elements of the CLBranchStack which is used for control flow analysis to compute maximum depth of operand stack for a method.
CLCodeAttribute	Representation of Code_attribute structure (JVM Spec Section 4.8.2).
CLComparisonInstruction	Representation for COMPARISON instructions.
CLConstantClassInfo	Representation of CONSTANT_Class_info structure (JVM Spec Section 4.5.1).
CLConstantDoubleInfo	Representation of CONSTANT_Double_info structure (JVM Spec Section 4.5.5).
CLConstantFieldRefInfo	Representation of CONSTANT_Fieldref_info structure (JVM Spec Section 4.5.2).
CLConstantFloatInfo	Representation of CONSTANT_Float_info structure (JVM Spec Section 4.5.4).
CLConstantIntegerInfo	Representation of CONSTANT_Integer_info structure (JVM Spec Section 4.5.4).

CLConstantInterfaceMethodRefInfo	Representation of CONSTANT_InterfaceMethodref_info structure (JVM Spec Section 4.5.2).
CLConstantLongInfo	Representation of CONSTANT_Long_info structure (JVM Spec Section 4.5.5).
CLConstantMemberRefInfo	Abstract super class of CONSTANT_Fieldref_info, CONSTANT_Methodref_info, CONSTANT_InterfaceMethodref_info structures (JVM Spec Section 4.5.2).
CLConstantMethodRefInfo	Representation of CONSTANT_Methodref_info structure (JVM Spec Section 4.5.2).
CLConstantNameAndTypeInfo	Representation of CONSTANT_NameAndType_info structure (JVM Spec Section 4.5.6).
CLConstantPool	Representation of a class' constant_pool table (JVM Spec Section 4.5).
CLConstants	Constants used within CL*.java files.
CLConstantStringInfo	Representation of CONSTANT_String_info structure (JVM Spec Section 4.5.3).
CLConstantUtf8Info	Representation of CONSTANT_Utf8_info structure (JVM Spec Section 4.5.7).
CLConstantValueAttribute	Representation of ConstantValue_attribute structure (JVM Spec Section 4.8.2).
CLConversionInstruction	Representation for CONVERSION instructions.
CLCPInfo	Representation of cp_info structure (JVM Spec Section 4.5).
CLDeprecatedAttribute	Representation of Deprecated_attribute structure (JVM Spec Section 4.8.14).
CLElementValue	Representation of element_value union (JVM Spec Section 4.8.15.1).
CLElementValuePair	Representation of the element_value_pairs table entry (JVM Spec Section 4.8.15).
CLEmitter	This class provides a high level interface for creating (in-memory and file based) representation of Java classes.

CLEnclosingMethodAttribute	Representation of EnclosingMethod_attribute structure (JVM Spec Section 4.8.6).
CLException	Representation of an exception handler.
CLExceptionInfo	Representation of exception_table entry structure (JVM Spec Section 4.8.3).
CLExceptionsAttribute	Representation of Exceptions_attribute structure (JVM Spec Section 4.8.4).
CLFieldInfo	Representation of field_info structure (JVM Spec Section 4.6).
CLFieldInstruction	Representation for FIELD instructions.
CLFile	Representation of the ClassFile structure (JVM Spec Section 4.2).
CLFlowControlInstruction	Representation for FLOW_CONTROL1, FLOW_CONTROL2, FLOW_CONTROL3 and FLOW_CONTROL4 instructions.
CLInnerClassesAttribute	Representation of InnerClasses_attribute structure (JVM Spec Section 4.8.5).
CLInnerClassInfo	Representation of classes table entry structure (JVM Spec Section 4.8.5).
CLInputStream	Inherits from java.io.DataInputStream and provides an extra function for reading unsigned int from the input stream, which is required for reading Java class files.
CLInsInfo	This class stores static information about a JVM instruction.
CLInstruction	Representation of a JVM instruction.
CLLineNumberInfo	Representation of line_number_table entry structure (JVM Spec Section 4.8.11).
CLLineNumberTableAttribute	Representation of LineNumberTable_attribute structure (JVM Spec Section 4.8.11).
CLLoadStoreInstruction	Representation for LOAD_STORE1, LOAD_STORE2, LOAD_STORE3 and LOAD_STORE4 instructions.
CLLocalVariableInfo	Representation of local_variable_table entry structure (JVM Spec Section 4.8.12).

CLLocalVariableTableAttribute	Representation of LocalVariableTable_attribute structure (JVM Spec Section 4.8.12).
CLLocalVariableTypeInfo	Representation of local_variable_type_table entry structure (JVM Spec Section 4.8.13).
CLLocalVariableTypeTableAttribute	Representation of LocalVariableTypeTable_attribute structure (JVM Spec Section 4.8.12).
CLMemberInfo	Abstraction of field_info and method_info structures (JVM Spec Section 4.6, 4.7).
CLMethodInfo	Representation of method_info structure (JVM Spec Section 4.7).
CLMethodInstruction	Representation for METHOD1 and METHOD2 instructions.
CLMiscInstruction	Representation for MISC instructions.
CLObjectInstruction	Representation for OBJECT instructions.
CLOutputStream	Inherits from java.out.DataOutputStream and provides an extra function for writing unsigned int to the output stream, which is required for writing Java class files.
CLParameterAnnotationInfo	Representation of parameter_annotations_table entry structure (JVM Spec Section 4.8.17).
CLPath	This class can be used to locate and load system, extension, and user-defined class files from directories and zip (jar) files.
CLRuntimeInvisibleAnnotationsAttribute	Representation of RuntimeInvisibleAnnotations_attribute structure (JVM Spec Section 4.8.16).
CLRuntimeInvisibleParameterAnnotationsAttribute	Representation of RuntimeInvisibleParameterAnnotations_attribute structure (JVM Spec Section 4.8.18).
CLRuntimeVisibleAnnotationsAttribute	Representation of RuntimeVisibleAnnotations_attribute structure (JVM Spec Section 4.8.15).
CLRuntimeVisibleParameterAnnotationsAttribute	Representation of RuntimeVisibleParameterAnnotations_attribute structure (JVM Spec Section 4.8.17).
CLSignatureAttribute	Representation of Signature_attribute

	structure (JVM Spec Section 4.8.8).
CLSourceDebugExtensionAttribute	Representation of SourceDebugExtension_attribute structure (JVM Spec Section 4.8.10).
CLSourceFileAttribute	Representation of SourceFile_attribute structure (JVM Spec Section 4.8.9).
CLStackInstruction	Representation for STACK instructions.
CLSyntheticAttribute	Representation of Synthetic_attribute structure (JVM Spec Section 4.8.7).
CompilationUnitContext	The compilation unit context is always the outermost context, and is where imported types and locally defined types (classes) are declared.
Constructor	A Constructor knows its JVM descriptor.
Context	A Context encapsulates the environment in which an AST is analyzed.
Conversions	A 2D table of conversions, from one type to another.
Field	A Field knows its type.
I2C	Converting from an int to a char requires an I2C instruction.
Identity	The identity conversion requires no run-time code.
JArrayExpression	The AST for an array indexing operation.
JArrayInitializer	The AST node for an array initializer.
JAssignment	The AST node for an assignment statement.
JAssignOp	The AST node for an assignment (=) expression.
JAST	The abstract superclass of all nodes in the abstract syntax tree (AST).
JavaCCMain	Driver class for j-- compiler using JavaCC front-end.
JavaCCParser	Parser generated by JavaCC.
JavaCCParser.JJCalls	
JavaCCParserTokenManager	

JBinaryExpression	The AST node for a binary expression.
JBlock	The AST node for a block, which delimits a nested level of scope.
JBooleanBinaryExpression	Most binary expressions that return booleans can be recognized by their syntax.
JCastOp	The AST for an cast expression, which has both a cast (a type) and the expression to be cast.
JClassDeclaration	A class declaration has a list of modifiers, a name, a super class and a class block; it distinguishes between instance fields and static (class) fields for initialization, and it defines a type.
JComparison	The AST node for a comparison expression.
JCompilationUnit	The abstract syntax tree (AST) node representing a compilation unit, and so the root of the AST.
JConstructorDeclaration	The AST node for a constructor declaration.
JEmptyStatement	The (dummy) AST node for representing the empty statement.
JEqualOp	The AST node for an equality (==) expression.
JExpression	The AST node for an expression.
JFieldDeclaration	The AST node for a field declaration.
JFieldSelection	The AST node for a field selection operation.
JFormalParameter	The AST node for a formal parameter declaration.
JGreaterThanOp	The AST node for a greater-than (>) expression.
JIfStatement	The AST node for an if-statement.
JInstanceOfOp	The AST node for an instanceof expression, having two arguments: an expression and a reference type.
JLessEqualOp	The AST node for a less-than-or-equal-to (<=) expression.
JLiteralChar	The AST node for a char literal.
JLiteralFalse	The AST node for the boolean "false"

	literal.
JLiteralInt	The AST node for an int literal.
JLiteralNull	The AST node for the null literal.
JLiteralString	The AST node for a string literal.
JLiteralTrue	The AST node for the boolean "true" literal.
JLogicalAndOp	The AST node for a logical AND (&&) expression.
JLogicalNotOp	The AST node for a logical NOT (!) expression.
JMessageExpression	The AST node for a message expression that has a target, optionally an ambiguous part, a message name, and zero or more actual arguments.
JMethodDeclaration	The AST node for a method declaration.
JMultiplyOp	The AST node for a multiplication (*) expression.
JNegateOp	The AST node for a unary negation (-) expression.
JNewArrayOp	The AST node for a "new" array operation.
JNewOp	The AST node for a "new" expression.
JPlusAssignOp	The AST node for a += expression.
JPlusOp	The AST node for a plus (+) expression.
JPostDecrementOp	The AST node for an expr--.
JPreIncrementOp	The AST node for a ++expr expression.
JReturnStatement	The AST node for a return-statement.
JStatement	The AST node for a statement (includes expressions).
JStatementExpression	The AST node for an expression that appears as a statement.
JStringConcatenationOp	The AST node for a string concatenation operation.
JSubtractOp	The AST node for a subtraction (-) expression.

JSuper	The AST node for a "super" expression.
JSuperConstruction	The AST node for a super(...) constructor.
JThis	The AST for a "this" expression.
JThisConstruction	The AST node for a this(...) constructor.
JUnaryExpression	The AST node for a unary expression.
JVariable	The AST node for an identifier used as a primary expression.
JVariableDeclaration	The AST node for a local variable declaration.
JVariableDeclarator	The AST node for a variable declarator, which declares a name, its type and (possibly) provides an initialization.
JWhileStatement	The AST node for a while-statement.
JWildExpression	The AST node for a "wild" expression.
LocalContext	A local context is a context (scope) in which local variables (including formal parameters) can be declared.
LocalVariableDefn	The definition for a local variable (including formal parameters).
LookaheadScanner	A lexical analyzer for j-- that interfaces with the hand-written parser (Parser.java).
Main	Driver class for j-- compiler using hand-written front-end.
Member	A wrapper for members (eg Fields, Methods, Constructors) in the Java API.
Method	A Method knows its descriptor (its signature in JVM format), and its return type.
MethodContext	A method context is where formal parameters are declared.
NarrowReference	A narrowing conversion on a reference type requires a run-time check on the type.
NBasicBlock	Representation of a block within a control flow graph.
NControlFlowGraph	Representation of a control flow graph

	(cfg) for a method.
NEmitter	A class for generating native SPIM code.
NGraphRegisterAllocator	Implements register allocation using graph coloring algorithm.
NHIRALoad	HIR instruction representing JVM array load instructions.
NHIRArithmetic	HIR instruction corresponding to the JVM arithmetic instructions.
NHIRAStore	HIR instruction representing JVM array store instructions.
NHIRConditionalJump	HIR instruction representing an conditional jump instructions in JVM.
NHIRGetField	HIR instruction representing JVM (get) field instructions.
NHIRGoto	HIR instruction representing an unconditional jump instruction in JVM.
NHIRInstruction	High-level intermediate representation (HIR) of a JVM instruction.
NHIRIntConstant	HIR instruction corresponding to the JVM instructions representing integer constants.
NHIRInvoke	HIR instruction representing method invocation instructions in JVM.
NHIRLoadLocal	HIR instruction representing a formal parameter.
NHIRLocal	HIR instruction representing a local (not formal) variable.
NHIRNewArray	HIR instruction representing JVM array creation instructions.
NHIRPhiFunction	HIR instruction representing phi functions.
NHIRPutField	HIR instruction representing JVM (put) field instructions.
NHIRReturn	HIR instruction representing a JVM return instruction.
NHIRStringConstant	HIR instruction corresponding to the JVM instructions representing string constants.
NInterval	A lifetime interval, recording the interval of LIR code for which the corresponding virtual register contains a useful value.
NLinearRegisterAllocator	Implements the Linear Scan register allocation algorithm.

NLIRALoad	LIR instruction representing JVM array load instructions.
NLIRArithmetic	LIR instruction corresponding to the JVM arithmetic instructions.
NLIRAStore	LIR instruction representing JVM array store instructions.
NLIRConditionalJump	LIR instruction representing an conditional jump instructions in JVM.
NLIRGetField	LIR instruction representing JVM (get) field instructions.
NLIRGoto	LIR instruction representing an unconditional jump instruction in JVM.
NLIRInstruction	Low-level intermediate representation (LIR) of a JVM instruction.
NLIRIntConstant	LIR instruction corresponding to the JVM instructions representing integer constants.
NLIRInvoke	LIR instruction representing method invocation instructions in JVM.
NLIRLoad	LIR instruction representing a load from memory to register.
NLIRLoadLocal	LIR instruction representing a formal parameter.
NLIRMove	LIR move instruction.
NLIRNewArray	LIR instruction representing JVM array creation instructions.
NLIRPhiFunction	LIR instruction representing phi functions.
NLIRPutField	LIR instruction representing JVM (put) field instructions.
NLIRReturn	HIR instruction representing a JVM return instruction.
NLIRStore	LIR instruction representing a store from a register to memory.
NLIRStringConstant	LIR instruction corresponding to the JVM instructions representing string constants.
NNaiveRegisterAllocator	Implemets a naive register allocation method.
NPhysicalRegister	An abstraction for a physical (SPIM) register.
NRange	A liveness range (for an interval).
NRegister	An abstraction for a (virtual or physical)

	register.
NRegisterAllocator	A register allocator maps virtual registers (from LIR code) to physical registers on the target machine.
NTuple	A tuple representation of a JVM instruction.
NVirtualRegister	An abstraction for a virtual register.
Parser	A recursive descent parser that, given a lexical analyzer (a LookaheadScanner), parses a Java compilation unit (program file), taking tokens from the LookaheadScanner, and produces an abstract syntax tree (AST) for it.
PrettyPrinter	A utility class that allows pretty (indented) printing to STDOUT.
Scanner	A lexical analyzer for j--, that has no backtracking mechanism.
SimpleCharStream	An implementation of interface CharStream, where the stream is assumed to contain only ASCII characters (without unicode processing).
Token	Describes the input token stream.
TokenInfo	A representation of tokens returned by the lexical analyzer method, getNextToken().
Type	For representing j-- types.
TypeName	Any reference type that can be denoted as a (possibly qualified) identifier.
TypeNameDefn	A definition of a type name.
UnBoxing	Unboxing requires invoking the appropriate conversion method from the (Java) API.
Util	This class defines helper functions.

Enum Summary

Enum	Description
CLConstants.Category	We classify the JVM instructions into the following categories.

InstructionType	The types of possible uses.
OffsetFrom	The types of stack pointers.
TokenKind	An enum of token kinds.

Exception Summary

Exception	Description
ParseException	This exception is thrown when parse errors are encountered.

Error Summary

Error	Description
TokenMgrError	

Package jminusminus Description

This package contains the scanner, parser, AST node representations, bytecode emitter, JavaCC description file, and driver classes for the j-- compiler.

- Prev Package
- Next Package

- Frames
- No Frames

- All Classes
- Help