JLhs.java

```java
// Copyright 2013 Bill Campbell, Swami Iyer and Bahar Akbal-Delibas

package jminusminus;

/**
 * The type of any expression that can appear on the lhs of an assignment
 * statement, i.e., JVariable, JFieldSelection, JArrayExpression.
 */

interface JLhs {

    /**
     * Analyze the lhs of an assignment. This is very much like analyze() but
     * perhaps a little more selective here and there.
     * 
     * @param context
     *            context in which names are resolved.
     * @return the analyzed (and possibly rewritten) AST subtree.
     */

    public JExpression analyzeLhs(Context context);

    /**
     * The up front code necessary for implementing an assignment; it generates
     * code to load onto the stack any part of the lhs variable that must be
     * there. For example, in a[i] = x, code must be generated to load the array
     * (a) and the index (i).
     * 
     * @param output
     *            the code emitter (basically an abstraction for producing the
     *            .class file).
     */

    public void codegenLoadLhsLvalue(CLEmitter output);

    /**
     * Generate the code required for loading an Rvalue for this variable, as in
     * a +=.
     * 
     * @param output
     *            the code emitter (basically an abstraction for producing the
     *            .class file).
     */

    public void codegenLoadLhsRvalue(CLEmitter output);

    /**
     * Generate the code required for duplicating the Rvalue that is on the
     * stack becuase it is to be used in a surrounding expression, as in a[i] =
     * x = <expr> or x = y--.
     * 
     * @param output
     *            the code emitter (basically an abstraction for producing the
     *            .class file).
     */

    public void codegenDuplicateRvalue(CLEmitter output);

    /**
     * Generate the code required for doing the actual assignment.
     * 
     * @param output
     *            the code emitter (basically an abstraction for producing the
     *            .class file).
     */

    public void codegenStore(CLEmitter output);

}

```