

The MaSH Programming Language At the Arrays Level

Andrew Rock

School of Information and Communication Technology

Griffith University

Nathan, Queensland, 4111, Australia

a.rock@griffith.edu.au

June 14, 2010

1 Introduction

This document defines the MaSH programming language at its arrays level. The arrays level adds the array data type.

This document only describes what is different or new with respect to the lower MaSH levels.

2 Lexical syntax

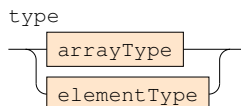
Same as for the lower levels.

3 Context-free grammar

Most of the grammar as presented up to the methods level is unchanged, except for the parts relating data types, variable and constant initialisation and expressions.

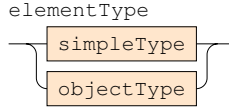
3.1 Types

The data types are now divided into the array types and the types that are permitted to be the elements of arrays. Note that arrays can be elements of arrays, by using multi-dimensional arrays.



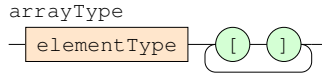
3.1.1 Element types

The element types include all of the data types except arrays – all of the simple types and all of the object types.



3.1.2 Array types

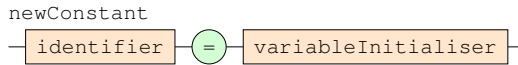
An array type is written as the type of the element, followed by a pair of brackets for each dimension of the array.



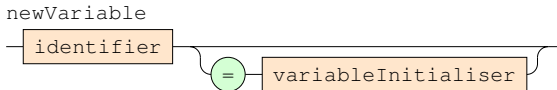
3.2 New constants and variables

The syntax for the introductions of new variables and constants changes because there is a special kind of expression for arrays that may only be used in variable and constant declarations.

3.2.1 New constants

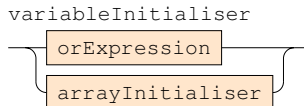


3.2.2 New variables



3.2.3 Variable initialisers

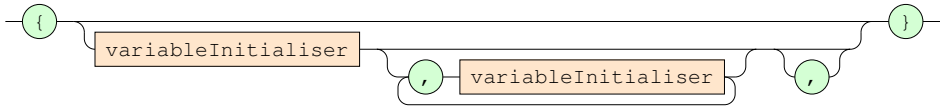
Variable initialisers are either expressions (as before) or array initialisers.



3.2.4 Array initialisers

An array initialiser is a list of comma separated variable initialisers, enclosed in braces. This creates a new array of these values, just big enough to hold those values. There can be an extra comma after the last variable initialiser in the list, and it is ignored.

arrayInitialiser



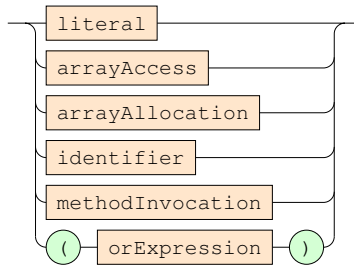
3.3 Expressions

3.3.1 Atomic expressions

There are two new kinds of atomic expressions:

- An array allocation represents a new array, allocated in memory and with all its elements equaling zero (or some value like zero that makes sense for the type of element).
- An array access is an expression that refers to an element of an array.

atomicExpression



3.3.2 Array allocations

An array allocation is written with the keyword `new` followed by the element type and then a pair of brackets for each dimension of the array. At least one of the leftmost pairs of brackets must have an integral value in them that declares the size of the array.

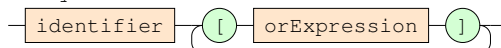
arrayAllocation



3.3.3 Array accesses

An array access starts with the name of the array variable followed by pairs of brackets, each containing the integral index number of the element being selected along each of the arrays dimensions.

arrayAccess



3.4 Assignments

3.4.1 Left-hand sides

The only change to assignments is that the left-hand sides may be array accesses or just identifiers.

