1. Title Page

Author: Amir Yousef

Date: December 7, 2013

Course Number: COP 4020

Semester Number: Fall 2013

Instructor Name: Chris Lacher

School Name: Florida State University

Assignment Name: Presentation and Paper

Assignment Title: Describe the Caml programming language

Assignment Summary: This assignment is describing the Caml programming language, and the assignment consists of following five sections:

Title Page: The cover page of the assignment

Introduction: The history of the Caml programming language implementation

Paper Body: The core of the Caml programming language including: Caml Variables, Caml Data Types, Caml Functions, and Caml Lexical Analysis

Conclusion: The advantages and disadvantages of the Caml programming language

References: The reference to each paper and web site that is consulted by the assignment

Embedded link to the Oral Personal Introduction: http://youtu.be/qUBPhkw0TWQ

Embedded link to the Technical Presentation: http://youtu.be/7-4EB7u76Mw

2. Introduction

The history of Caml programming language implementation was originally started on 1987. The Caml programming language is one of the widely used programming languages, because the compilers are all open source software, available free with no charge, and it supports functional, imperative, and object-oriented programming styles.

"Caml (Categorical Abstract Machine Language) is a dialect of the ML programming language family, developed at INRIA. Caml is statically typed, strictly evaluated, and uses automatic memory management." [According to http://en.wikipedia.org/wiki/Caml]

The first Caml implementation on 1987, it was successful but requires more memory and CPU. The Caml Light was implemented on 1990 by Xavier Leroy based on interpreter written in C to improve the first Caml implementation. The Caml Special Light was implemented on 1995 by Xavier Leroy to improve the Caml Light by adding a native code compiler to match or exceed the performances of the existing functional languages compilers, and enabled Caml to have a competitive performance with imperative programming languages. The first Objective Caml language was implemented on 1996, renamed to OCaml in 2011. The OCaml was the first language has the object-oriented programming with ML-style static typing.

"OCaml supports many advanced OO programming idioms (type-parametric classes, binary methods, mytype specialization) in a statically type-safe way. While these idioms require run-time type checks in other OO languages such as C++ and Java."
[According to http://caml.inria.fr/about/history.en.html]

3. Paper Body

3.1 Caml Variables

[According to The Caml Light manual: http://caml.inria.fr/pub/docs/manual-caml-light/]

Caml programming language has many variable types such as {integers, floats, characters, character strings, tuples, records, arrays, and variants}. Integer values are integer numbers from -1073741824 to 1073741823. Floating-point values are numbers in floating-point representation. Character values are represented as 8-bit integers between 0 and 255. Character codes between 0 and 127 are interpreted by the ASCII standard. Character strings values are finite sequences of characters up to 65535 characters. Tuples of values are written in the form of (v_1, ..., v_n), standing for the n-tuple of values v_1 to v_n. Record values are labeled tuples of values. The record value written { label_1 = v_1; ...; label_n = v_n } associates the value v_i to the record label label_i, for i = 1 ... n. Arrays are finite, variable-sized sequences of values of the same type. Arrays of length up to 16383 elements. Variant values are either a constant constructor, or a pair of a non-constant constructor and a value.

3.2 Caml Data Types

[According to. Introduction to Caml. http://www.cs.jhu.edu/~scott/pl/lectures/caml-intro.html]

Caml programming language has a built in data types such as lists and tuples. Lists are a list of Caml values, automatically allocates space for the list and puts in the elements. All elements of a list must be of the same type. Caml is garbage collected, no explicit de-allocation is needed. Caml language dealing very easy with Lists/Tuples more than any other language. Notice in the following examples how the type either int list or string in the list is inferred automatically without need to identify the type of the list:

```
#[1;1+1;3];;
- : int list = [1; 2; 3]
["a"; "b"; "c"];;
#- : string list = ["a"; "b"; "c"]
```

Caml can also do list operations such as join lists together “appending lists” .

```
let x = [1; 2; 3];;
#x : int list = [1; 2; 3]
let y = [4; 5; 6];;
#y : int list = [4; 5; 6]
x @ y;;          (* appending lists x, y *)
#- : int list = [1; 2; 3; 4; 5; 6]
```

Tuples are fixed-length lists, but the fields may be different types for example:

```
let x = (1, 2, 3, "am", "is", "are");;
#x : int * int * int * string * string * string = 1, 2, 3, "am", "is", "are"

match x with (a, b, c, d, e, f) -> a;;     (* match will return the value matched element a from x*)
#- : int = 1

match x with (a, b, c, d, e, f) -> f;;     (* match will return the value matched element f from x*)
#- : string = "are"
```

3.3 Caml Functions

Caml functions can be defined anywhere, no return statement, the functions can be passed to and returned from functions, the value of the whole expression is returned for example:

```
# ((function x -> x+2) 3) + 5;;
- : int = 10
```

Function types are grammatically printed and are always printed as <fun> for example:

```
# let f = (function x -> x+2);;
f : int -> int = <fun>
```

Caml functions can take only one argument, and this is one of the weaknesses of Caml programming language, because multiple argument functions are not built in and we need to use Currying to define them for example:

```
# let rec f x y = if y=0 or y=x then 1 else f (x-1) y + f (x-1) (y-1);;
 f : int -> int -> int = <fun>
# f 8 2;;
- : int = 28
```

Now, f “defined function” takes only one integer and returns a function for example:

```
let f2 = f  2;;
f2 2;;
f2 : int -> int = <fun>
#- : int = 1
```

3.4 Caml Lexical Analysis

Caml programming language ignores any blanks. Comments are introduced by the two characters (* and *) and they are treated as blank characters. In Caml any comment, space, newline, horizontal tabulation, carriage return, line feed and form feed are considered as blank characters. Identifiers are sequences of letters, digits and underscore character, starting with a letter. Letters are the fifty-two lowercase and uppercase letters from the ASCII. Identifiers cannot contain two adjacent (after each other) underscore characters. All of the identifier characters are meaningful. Integer literal is a sequence of one or more digits, optionally preceded by a minus sign. By default, integer literals are in decimal (radix 10). Floating-point has an integer part, a decimal part and an exponent part. The integer part is a sequence of one or more digits, optionally preceded by a minus sign. The decimal part is a decimal point followed by zero, one or more digits. The exponent part is the character e or E followed by an optional + or - sign, followed by one or more digits. The decimal part or the exponent part can be omitted, but not both to avoid ambiguity with integer literals. Character literals are delimited by back quote characters. String literals are delimited by double quote characters.

These identifiers and characters are reserved as keywords, and cannot be used otherwise:

{ and as begin do done downto else end exception for fun function if in let match mutable not of or prefix rec then to try type value where while with # ! != & ( ) * *. + +. , - -. -> . .(/ /. : :: := ; ;; < <. <- <= <=. <> <>. = =. == > >. >= >=. @ [[|] ^ _ __ { | |] } ' }

[According to. Xavier Leroy. (1997). The Caml Light system release 0.73 Documentation and user's manual. http://caml.inria.fr/pub/distrib/caml-light-0.74//cl74refman.txt]

4. Conclusion

My conclusion is a comparison between the advantages and the disadvantages of Caml language.

The advantages of the Caml programming language are:

1. Caml is one of the widely used programming languages, the compilers are all open source software, available free with no charge, and it supports functional, imperative, and object-oriented programming styles.
2. Caml is statically typed, strictly evaluated, and uses automatic memory management, and incremental garbage collector.
3. The compiler recognizes how every variable is used, and derives the minimal necessary typing constraints from that (if you typed x=1;; then the compiler knows that x is an integer).
4. Caml has a type inference allows defining such operations without having to explicitly provide the type of their parameters and result.
5. Caml deal very easy with lists and tuples, can create lists by their own proper type, we don't identify the type, and it will be created and de-allocated automatically. Caml has operations such as append lists together and match lists elements.
6. Caml has a garbage collector, no allocate or free memory required, it's done automatically.

The disadvantages of Caml programming language:

1. Caml functions can take only one argument, and this is one of the weaknesses of Caml programming language, because multiple argument functions are not built in and we need to use Currying to define them, in my opinion that one of the biggest issues of Caml.
2. Caml is a great language if you want to solve an algorithmic problem that does not depend too much on having to interact with advanced standard modules like XML or networking.

3. Caml is not popular programming language like C.

4. Caml was just behind C and C++ for the speed of execution.

5. Caml is good for education purpose more than for make applications.

6. Caml application is limited not like java application that can be used in any computer and can be spread over internet very easily.

7. Finally the standard library, Caml has almost everything such as headers, directives, type definitions built in to make it easier but at the same time we are missing all information about these parts and it's better if we know them in my opinion.

References

These are my references:

The Caml Language: http://caml.inria.fr/about/history.en.html

Wikipedia: http://en.wikipedia.org/wiki/Caml

Introduction to Caml: http://www.cs.jhu.edu/~scott/pl/lectures/caml-intro.html

The Caml Light manual: http://caml.inria.fr/pub/docs/manual-caml-light/

Xavier Leroy. (1997). The Caml Light system release 0.73 Documentation and user's manual: http://caml.inria.fr/pub/distrib/caml-light-0.74//cl74refman.txt

"Thank you for your time reading my paper"

Amir Yousef