

# Reference Manual for the Objective-C Language

This appendix presents a formal grammar for the Objective-C extensions to the C language. As the Objective-C language is implemented for the OPENSTEP development environment. It adds to the grammar for ANSI standard C found in Appendix A of *The C Programming Language* (second edition, 1988) by Brian W. Kernighan and Dennis M. Ritchie, published by Prentice Hall, and should be read in conjunction with that book.

The Objective-C extensions introduce some new symbols (such as *class-interface*), but also make use of symbols (such as *function-definition*) that are explained in the standard C grammar. The symbols mentioned but not explained here are listed below:

## Undefined Symbols

274939\_TableHeadRule.eps ↵

compound statement	identifier
constant	parameter-type-list
declaration	string
declaration-list	struct-declaration-list
enum-specifier	struct-or-union
expression	typedef-name
function-definition	type-name

Of these, *identifier* and *string* are undefined terminal symbols. Objective-C adds no undefined terminal symbols of its own.

Two notational conventions used here differ from those used in *The C Programming Language*:

- Literal symbols are shown in **bold** type.
- Brackets enclose optional elements and are in *italic* type. Literal brackets, like other literal symbols, are non-italic and bold.

Otherwise, this appendix follows the conventions of the C reference manual. Each part of the grammar consists of a symbol followed by a colon and an indented list of mutually-exclusive possibilities for expanding the symbol. For example:

*receiver:*  
***expression***

*class-name*

**super**

However, there is an exception: Even though they're not mutually exclusive, the constituents of classes, categories, and protocols are listed on separate lines to clearly show the ordering of elements. For example:

*protocol-declaration:*

**@protocol** *protocol-name*

[ *protocol-reference-list* ]

[ *interface-declaration-list* ]

**@end**

This exception to the general rule is easily recognized since each list terminates with **@end**.

There are just four entry points where the Objective-C language modifies the rules defined for standard C:

- External declarations
- Type specifiers
- Type qualifiers
- Primary expressions

This appendix is therefore divided into four sections corresponding to these points. Where a rule in the standard C grammar is modified by an Objective-C extension, the entire rule is repeated in its modified form.

## External Declarations

*external-declaration:*

*function-definition*  
*declaration*  
*class-interface*  
*class-implementation*  
*category-interface*  
*category-implementation*  
*protocol-declaration*  
*class-declaration-list*

*class-interface:*

```
@interface class-name [: superclass-name ]  
[ protocol-reference-list ]  
[ instance-variables ]  
[ interface-declaration-list ]  
@end
```

*class-implementation:*

```
@implementation class-name [: superclass-name ]  
[ instance-variables ]  
[ implementation-definition-list ]  
@end
```

*category-interface:*

```
@interface class-name ( category-name )  
[ protocol-reference-list ]  
[ interface-declaration-list ]  
@end
```

*category-implementation:*

**@implementation** *class-name* ( *category-name* )  
[ *implementation-definition-list* ]  
**@end**

*protocol-declaration:*  
**@protocol** *protocol-name*  
[ *protocol-reference-list* ]  
[ *interface-declaration-list* ]  
**@end**

*class-declaration-list:*  
**@class** *class-list* ;

*class-list:*  
*class-name*  
*class-list* , *class-name*

*protocol-reference-list:*  
< *protocol-list* >

*protocol-list:*  
*protocol-name*  
*protocol-list* , *protocol-name*

*class-name:*  
*identifier*

*superclass-name:*  
*identifier*

*category-name:*  
*identifier*

*protocol-name:*  
*identifier*

*instance-variables:*  
{ [ *visibility-specification* ] *struct-declaration-list* [ *instance-variables* ] }

*visibility-specification:*  
**@private**  
**@protected**  
**@public**

*interface-declaration-list:*  
*declaration*  
*method-declaration*  
*interface-declaration-list declaration*  
*interface-declaration-list method-declaration*

*method-declaration:*  
*class-method-declaration*  
*instance-method-declaration*

*class-method-declaration:*  
+ [ *method-type* ] *method-selector* ;

*instance-method-declaration:*  
- [ *method-type* ] *method-selector* ;

*implementation-definition-list:*

*function-definition*  
*declaration*  
*method-definition*  
*implementation-definition-list* *function-definition*  
*implementation-definition-list* *declaration*  
*implementation-definition-list* *method-definition*

*method-definition:*  
*class-method-definition*  
*instance-method-definition*

*class-method-definition:*  
+ [ *method-type* ] *method-selector* [ *declaration-list* ] *compound-statement*

*instance-method-definition:*  
- [ *method-type* ] *method-selector* [ *declaration-list* ] *compound-statement*

*method-selector:*  
*unary-selector*  
*keyword-selector* [ , ... ]  
*keyword-selector* [ , *parameter-type-list* ]

*unary-selector:*  
*selector*

*keyword-selector:*  
*keyword-declarator*  
*keyword-selector* *keyword-declarator*

*keyword-declarator:*

*: [method-type] identifier*  
*selector : [method-type] identifier*

*selector:*  
*identifier*

*method-type:*  
*( type-name )*

## Type Specifiers

*type-specifier:*

**void**

**char**

**short**

**int**

**long**

**float**

**double**

**signed**

**unsigned**

**id** *[ protocol-reference-list ]*

*class-name* *[ protocol-reference-list ]*

*struct-or-union-specifier*

*enum-specifier*

*typedef-name*

*struct-or-union-specifier:*

*struct-or-union* [ *identifier* ] { *struct-declaration-list* }  
*struct-or-union* [ *identifier* ] { @defs ( *class-name* ) }  
*struct-or-union* *identifier*

## Type Qualifiers

*type-qualifier:*

**const**

**volatile**

*protocol-qualifier*

*protocol-qualifier:*

**in**

**out**

**inout**

**bycopy**

**byref**

**oneway**

## Primary Expressions

*primary-expression:*

*identifier*

*constant*

*string*

*( expression )*

**self**

*message-expression*

*selector-expression*

*protocol-expression*

*encode-expression*

*message-expression:*

[ *receiver message-selector* ]

*receiver:*

*expression*

*class-name*

**super**

*message-selector:*

*selector*

*keyword-argument-list*

*keyword-argument-list:*

*keyword-argument*

*keyword-argument-list keyword-argument*

*keyword-argument:*

*selector : expression*

*: expression*

*selector-expression:*

**@selector** ( *selector-name* )

*selector-name:*

*selector*

*keyword-name-list*

*keyword-name-list:*

*keyword-name*

*keyword-name-list keyword-name*

*keyword-name:*

*selector :*

*:*

*protocol-expression:*

**@protocol** ( *protocol-name* )

*encode-expression:*

**@encode** ( *type-name* )