// <https://docs.swift.org/swift-book/ReferenceManual/zzSummaryOfTheGrammar.html>

Lexical Structure

GRAMMAR OF WHITESPACE

*whitespace* → [whitespace-item](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_whitespace-item) [whitespace](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_whitespace) *opt*

*whitespace-item* → [line-break](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_line-break)

*whitespace-item* → [inline-space](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_inline-space)

*whitespace-item* → [comment](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_comment)

*whitespace-item* → [multiline-comment](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-comment)

*whitespace-item* → U+0000, U+000B, or U+000C

*line-break* → U+000A

*line-break* → U+000D

*line-break* → U+000D followed by U+000A

*inline-spaces* → [inline-space](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_inline-space) [inline-spaces](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_inline-spaces) *opt*

*inline-space* → U+0009 or U+0020

*comment* → **//** [comment-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_comment-text) [line-break](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_line-break)

*multiline-comment* → **/\*** [multiline-comment-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-comment-text) **\*/**

*comment-text* → [comment-text-item](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_comment-text-item) [comment-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_comment-text) *opt*

*comment-text-item* → Any Unicode scalar value except U+000A or U+000D

*multiline-comment-text* → [multiline-comment-text-item](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-comment-text-item) [multiline-comment-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-comment-text) *opt*

*multiline-comment-text-item* → [multiline-comment](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-comment)

*multiline-comment-text-item* → [comment-text-item](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_comment-text-item)

*multiline-comment-text-item* → Any Unicode scalar value except **/\*** or **\*/**

GRAMMAR OF AN IDENTIFIER

*identifier* → [identifier-head](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier-head) [identifier-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier-characters) *opt*

*identifier* → **`** [identifier-head](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier-head) [identifier-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier-characters) *opt* **`**

*identifier* → [implicit-parameter-name](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_implicit-parameter-name)

*identifier* → [property-wrapper-projection](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_property-wrapper-projection)

*identifier-list* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) | [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) **,** [identifier-list](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier-list)

*identifier-head* → Upper- or lowercase letter A through Z

*identifier-head* → **\_**

*identifier-head* → U+00A8, U+00AA, U+00AD, U+00AF, U+00B2–U+00B5, or U+00B7–U+00BA

*identifier-head* → U+00BC–U+00BE, U+00C0–U+00D6, U+00D8–U+00F6, or U+00F8–U+00FF

*identifier-head* → U+0100–U+02FF, U+0370–U+167F, U+1681–U+180D, or U+180F–U+1DBF

*identifier-head* → U+1E00–U+1FFF

*identifier-head* → U+200B–U+200D, U+202A–U+202E, U+203F–U+2040, U+2054, or U+2060–U+206F

*identifier-head* → U+2070–U+20CF, U+2100–U+218F, U+2460–U+24FF, or U+2776–U+2793

*identifier-head* → U+2C00–U+2DFF or U+2E80–U+2FFF

*identifier-head* → U+3004–U+3007, U+3021–U+302F, U+3031–U+303F, or U+3040–U+D7FF

*identifier-head* → U+F900–U+FD3D, U+FD40–U+FDCF, U+FDF0–U+FE1F, or U+FE30–U+FE44

*identifier-head* → U+FE47–U+FFFD

*identifier-head* → U+10000–U+1FFFD, U+20000–U+2FFFD, U+30000–U+3FFFD, or U+40000–U+4FFFD

*identifier-head* → U+50000–U+5FFFD, U+60000–U+6FFFD, U+70000–U+7FFFD, or U+80000–U+8FFFD

*identifier-head* → U+90000–U+9FFFD, U+A0000–U+AFFFD, U+B0000–U+BFFFD, or U+C0000–U+CFFFD

*identifier-head* → U+D0000–U+DFFFD or U+E0000–U+EFFFD

*identifier-character* → Digit 0 through 9

*identifier-character* → U+0300–U+036F, U+1DC0–U+1DFF, U+20D0–U+20FF, or U+FE20–U+FE2F

*identifier-character* → [identifier-head](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier-head)

*identifier-characters* → [identifier-character](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier-character) [identifier-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier-characters) *opt*

*implicit-parameter-name* → **$** [decimal-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digits)

*property-wrapper-projection* → **$** [identifier-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier-characters)

GRAMMAR OF A LITERAL

*literal* → [numeric-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_numeric-literal) | [string-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_string-literal) | [boolean-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_boolean-literal) | [nil-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_nil-literal)

*numeric-literal* → **-***opt* [integer-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_integer-literal) | **-***opt* [floating-point-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_floating-point-literal)

*boolean-literal* → **true** | **false**

*nil-literal* → **nil**

GRAMMAR OF AN INTEGER LITERAL

*integer-literal* → [binary-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_binary-literal)

*integer-literal* → [octal-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_octal-literal)

*integer-literal* → [decimal-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-literal)

*integer-literal* → [hexadecimal-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_hexadecimal-literal)

*binary-literal* → **0b** [binary-digit](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_binary-digit) [binary-literal-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_binary-literal-characters) *opt*

*binary-digit* → Digit 0 or 1

*binary-literal-character* → [binary-digit](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_binary-digit) | **\_**

*binary-literal-characters* → [binary-literal-character](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_binary-literal-character) [binary-literal-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_binary-literal-characters) *opt*

*octal-literal* → **0o** [octal-digit](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_octal-digit) [octal-literal-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_octal-literal-characters) *opt*

*octal-digit* → Digit 0 through 7

*octal-literal-character* → [octal-digit](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_octal-digit) | **\_**

*octal-literal-characters* → [octal-literal-character](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_octal-literal-character) [octal-literal-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_octal-literal-characters) *opt*

*decimal-literal* → [decimal-digit](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digit) [decimal-literal-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-literal-characters) *opt*

*decimal-digit* → Digit 0 through 9

*decimal-digits* → [decimal-digit](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digit) [decimal-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digits) *opt*

*decimal-literal-character* → [decimal-digit](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digit) | **\_**

*decimal-literal-characters* → [decimal-literal-character](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-literal-character) [decimal-literal-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-literal-characters) *opt*

*hexadecimal-literal* → **0x** [hexadecimal-digit](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_hexadecimal-digit) [hexadecimal-literal-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_hexadecimal-literal-characters) *opt*

*hexadecimal-digit* → Digit 0 through 9, a through f, or A through F

*hexadecimal-literal-character* → [hexadecimal-digit](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_hexadecimal-digit) | **\_**

*hexadecimal-literal-characters* → [hexadecimal-literal-character](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_hexadecimal-literal-character) [hexadecimal-literal-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_hexadecimal-literal-characters) *opt*

GRAMMAR OF A FLOATING-POINT LITERAL

*floating-point-literal* → [decimal-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-literal) [decimal-fraction](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-fraction) *opt* [decimal-exponent](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-exponent) *opt*

*floating-point-literal* → [hexadecimal-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_hexadecimal-literal) [hexadecimal-fraction](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_hexadecimal-fraction) *opt* [hexadecimal-exponent](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_hexadecimal-exponent)

*decimal-fraction* → **.** [decimal-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-literal)

*decimal-exponent* → [floating-point-e](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_floating-point-e) [sign](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_sign) *opt* [decimal-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-literal)

*hexadecimal-fraction* → **.** [hexadecimal-digit](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_hexadecimal-digit) [hexadecimal-literal-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_hexadecimal-literal-characters) *opt*

*hexadecimal-exponent* → [floating-point-p](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_floating-point-p) [sign](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_sign) *opt* [decimal-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-literal)

*floating-point-e* → **e** | **E**

*floating-point-p* → **p** | **P**

*sign* → **+** | **-**

GRAMMAR OF A STRING LITERAL

*string-literal* → [static-string-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_static-string-literal) | [interpolated-string-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_interpolated-string-literal)

*string-literal-opening-delimiter* → [extended-string-literal-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_extended-string-literal-delimiter) *opt* **"**

*string-literal-closing-delimiter* → **"** [extended-string-literal-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_extended-string-literal-delimiter) *opt*

*static-string-literal* → [string-literal-opening-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_string-literal-opening-delimiter) [quoted-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_quoted-text) *opt* [string-literal-closing-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_string-literal-closing-delimiter)

*static-string-literal* → [multiline-string-literal-opening-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-string-literal-opening-delimiter) [multiline-quoted-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-quoted-text) *opt* [multiline-string-literal-closing-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-string-literal-closing-delimiter)

*multiline-string-literal-opening-delimiter* → [extended-string-literal-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_extended-string-literal-delimiter) **"""**

*multiline-string-literal-closing-delimiter* → **"""** [extended-string-literal-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_extended-string-literal-delimiter)

*extended-string-literal-delimiter* → **#** [extended-string-literal-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_extended-string-literal-delimiter) *opt*

*quoted-text* → [quoted-text-item](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_quoted-text-item) [quoted-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_quoted-text) *opt*

*quoted-text-item* → [escaped-character](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escaped-character)

*quoted-text-item* → Any Unicode scalar value except **"**, **\**, U+000A, or U+000D

*multiline-quoted-text* → [multiline-quoted-text-item](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-quoted-text-item) [multiline-quoted-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-quoted-text) *opt*

*multiline-quoted-text-item* → [escaped-character](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escaped-character)

*multiline-quoted-text-item* → Any Unicode scalar value except **\**

*multiline-quoted-text-item* → [escaped-newline](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escaped-newline)

*interpolated-string-literal* → [string-literal-opening-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_string-literal-opening-delimiter) [interpolated-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_interpolated-text) *opt* [string-literal-closing-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_string-literal-closing-delimiter)

*interpolated-string-literal* → [multiline-string-literal-opening-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-string-literal-opening-delimiter) [multiline-interpolated-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-interpolated-text) *opt* [multiline-string-literal-closing-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-string-literal-closing-delimiter)

*interpolated-text* → [interpolated-text-item](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_interpolated-text-item) [interpolated-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_interpolated-text) *opt*

*interpolated-text-item* → **\(** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **)** | [quoted-text-item](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_quoted-text-item)

*multiline-interpolated-text* → [multiline-interpolated-text-item](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-interpolated-text-item) [multiline-interpolated-text](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-interpolated-text) *opt*

*multiline-interpolated-text-item* → **\(** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **)** | [multiline-quoted-text-item](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_multiline-quoted-text-item)

*escape-sequence* → **\** [extended-string-literal-delimiter](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_extended-string-literal-delimiter)

*escaped-character* → [escape-sequence](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escape-sequence) **0** | [escape-sequence](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escape-sequence) **\** | [escape-sequence](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escape-sequence) **t** | [escape-sequence](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escape-sequence) **n** | [escape-sequence](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escape-sequence) **r** | [escape-sequence](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escape-sequence) **"** | [escape-sequence](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escape-sequence) **'**

*escaped-character* → [escape-sequence](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escape-sequence) **u** **{** [unicode-scalar-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_unicode-scalar-digits) **}**

*unicode-scalar-digits* → Between one and eight hexadecimal digits

*escaped-newline* → [escape-sequence](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_escape-sequence) [inline-spaces](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_inline-spaces) *opt* [line-break](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_line-break)

GRAMMAR OF OPERATORS

*operator* → [operator-head](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator-head) [operator-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator-characters) *opt*

*operator* → [dot-operator-head](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_dot-operator-head) [dot-operator-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_dot-operator-characters)

*operator-head* → **/** | **=** | **-** | **+** | **!** | **\*** | **%** | **<** | **>** | **&** | **|** | **^** | **~** | **?**

*operator-head* → U+00A1–U+00A7

*operator-head* → U+00A9 or U+00AB

*operator-head* → U+00AC or U+00AE

*operator-head* → U+00B0–U+00B1

*operator-head* → U+00B6, U+00BB, U+00BF, U+00D7, or U+00F7

*operator-head* → U+2016–U+2017

*operator-head* → U+2020–U+2027

*operator-head* → U+2030–U+203E

*operator-head* → U+2041–U+2053

*operator-head* → U+2055–U+205E

*operator-head* → U+2190–U+23FF

*operator-head* → U+2500–U+2775

*operator-head* → U+2794–U+2BFF

*operator-head* → U+2E00–U+2E7F

*operator-head* → U+3001–U+3003

*operator-head* → U+3008–U+3020

*operator-head* → U+3030

*operator-character* → [operator-head](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator-head)

*operator-character* → U+0300–U+036F

*operator-character* → U+1DC0–U+1DFF

*operator-character* → U+20D0–U+20FF

*operator-character* → U+FE00–U+FE0F

*operator-character* → U+FE20–U+FE2F

*operator-character* → U+E0100–U+E01EF

*operator-characters* → [operator-character](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator-character) [operator-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator-characters) *opt*

*dot-operator-head* → **.**

*dot-operator-character* → **.** | [operator-character](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator-character)

*dot-operator-characters* → [dot-operator-character](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_dot-operator-character) [dot-operator-characters](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_dot-operator-characters) *opt*

*binary-operator* → [operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator)

*prefix-operator* → [operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator)

*postfix-operator* → [operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator)

Types

GRAMMAR OF A TYPE

*type* → [function-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_function-type)

*type* → [array-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_array-type)

*type* → [dictionary-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_dictionary-type)

*type* → [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier)

*type* → [tuple-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_tuple-type)

*type* → [optional-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_optional-type)

*type* → [implicitly-unwrapped-optional-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_implicitly-unwrapped-optional-type)

*type* → [protocol-composition-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_protocol-composition-type)

*type* → [opaque-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_opaque-type)

*type* → [metatype-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_metatype-type)

*type* → [any-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_any-type)

*type* → [self-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_self-type)

*type* → **(** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type) **)**

GRAMMAR OF A TYPE ANNOTATION

*type-annotation* → **:** [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* **inout***opt* [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

GRAMMAR OF A TYPE IDENTIFIER

*type-identifier* → [type-name](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-name) [generic-argument-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-argument-clause) *opt* | [type-name](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-name) [generic-argument-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-argument-clause) *opt* **.** [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier)

*type-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

GRAMMAR OF A TUPLE TYPE

*tuple-type* → **(** **)** | **(** [tuple-type-element](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_tuple-type-element) **,** [tuple-type-element-list](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_tuple-type-element-list) **)**

*tuple-type-element-list* → [tuple-type-element](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_tuple-type-element) | [tuple-type-element](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_tuple-type-element) **,** [tuple-type-element-list](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_tuple-type-element-list)

*tuple-type-element* → [element-name](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_element-name) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) | [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

*element-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

GRAMMAR OF A FUNCTION TYPE

*function-type* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [function-type-argument-clause](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_function-type-argument-clause) **throws***opt* **->** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

*function-type-argument-clause* → **(** **)**

*function-type-argument-clause* → **(** [function-type-argument-list](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_function-type-argument-list) **...***opt* **)**

*function-type-argument-list* → [function-type-argument](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_function-type-argument) | [function-type-argument](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_function-type-argument) **,** [function-type-argument-list](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_function-type-argument-list)

*function-type-argument* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* **inout***opt* [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type) | [argument-label](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_argument-label) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation)

*argument-label* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

GRAMMAR OF AN ARRAY TYPE

*array-type* → **[** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type) **]**

GRAMMAR OF A DICTIONARY TYPE

*dictionary-type* → **[** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type) **:** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type) **]**

GRAMMAR OF AN OPTIONAL TYPE

*optional-type* → [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type) **?**

GRAMMAR OF AN IMPLICITLY UNWRAPPED OPTIONAL TYPE

*implicitly-unwrapped-optional-type* → [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type) **!**

GRAMMAR OF A PROTOCOL COMPOSITION TYPE

*protocol-composition-type* → [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier) **&** [protocol-composition-continuation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_protocol-composition-continuation)

*protocol-composition-continuation* → [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier) | [protocol-composition-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_protocol-composition-type)

GRAMMAR OF AN OPAQUE TYPE

*opaque-type* → **some** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

GRAMMAR OF A METATYPE TYPE

*metatype-type* → [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type) **.** **Type** | [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type) **.** **Protocol**

GRAMMAR OF AN ANY TYPE

*any-type* → **Any**

GRAMMAR OF A SELF TYPE

*self-type* → **Self**

GRAMMAR OF A TYPE INHERITANCE CLAUSE

*type-inheritance-clause* → **:** [type-inheritance-list](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-inheritance-list)

*type-inheritance-list* → [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier) | [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier) **,** [type-inheritance-list](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-inheritance-list)

Expressions

GRAMMAR OF AN EXPRESSION

*expression* → [try-operator](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_try-operator) *opt* [prefix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_prefix-expression) [binary-expressions](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_binary-expressions) *opt*

*expression-list* → [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) | [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **,** [expression-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression-list)

GRAMMAR OF A PREFIX EXPRESSION

*prefix-expression* → [prefix-operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_prefix-operator) *opt* [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression)

*prefix-expression* → [in-out-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_in-out-expression)

GRAMMAR OF AN IN-OUT EXPRESSION

*in-out-expression* → **&** [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

GRAMMAR OF A TRY EXPRESSION

*try-operator* → **try** | **try** **?** | **try** **!**

GRAMMAR OF A BINARY EXPRESSION

*binary-expression* → [binary-operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_binary-operator) [prefix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_prefix-expression)

*binary-expression* → [assignment-operator](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_assignment-operator) [try-operator](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_try-operator) *opt* [prefix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_prefix-expression)

*binary-expression* → [conditional-operator](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_conditional-operator) [try-operator](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_try-operator) *opt* [prefix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_prefix-expression)

*binary-expression* → [type-casting-operator](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_type-casting-operator)

*binary-expressions* → [binary-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_binary-expression) [binary-expressions](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_binary-expressions) *opt*

GRAMMAR OF AN ASSIGNMENT OPERATOR

*assignment-operator* → **=**

GRAMMAR OF A CONDITIONAL OPERATOR

*conditional-operator* → **?** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **:**

GRAMMAR OF A TYPE-CASTING OPERATOR

*type-casting-operator* → **is** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

*type-casting-operator* → **as** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

*type-casting-operator* → **as** **?** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

*type-casting-operator* → **as** **!** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

GRAMMAR OF A PRIMARY EXPRESSION

*primary-expression* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) [generic-argument-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-argument-clause) *opt*

*primary-expression* → [literal-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_literal-expression)

*primary-expression* → [self-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_self-expression)

*primary-expression* → [superclass-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_superclass-expression)

*primary-expression* → [closure-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_closure-expression)

*primary-expression* → [parenthesized-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_parenthesized-expression)

*primary-expression* → [tuple-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_tuple-expression)

*primary-expression* → [implicit-member-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_implicit-member-expression)

*primary-expression* → [wildcard-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_wildcard-expression)

*primary-expression* → [key-path-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_key-path-expression)

*primary-expression* → [selector-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_selector-expression)

*primary-expression* → [key-path-string-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_key-path-string-expression)

GRAMMAR OF A LITERAL EXPRESSION

*literal-expression* → [literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_literal)

*literal-expression* → [array-literal](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_array-literal) | [dictionary-literal](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_dictionary-literal) | [playground-literal](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_playground-literal)

*literal-expression* → **#file** | **#fileID** | **#filePath**

*literal-expression* → **#line** | **#column** | **#function** | **#dsohandle**

*array-literal* → **[** [array-literal-items](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_array-literal-items) *opt* **]**

*array-literal-items* → [array-literal-item](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_array-literal-item) **,***opt* | [array-literal-item](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_array-literal-item) **,** [array-literal-items](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_array-literal-items)

*array-literal-item* → [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression)

*dictionary-literal* → **[** [dictionary-literal-items](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_dictionary-literal-items) **]** | **[** **:** **]**

*dictionary-literal-items* → [dictionary-literal-item](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_dictionary-literal-item) **,***opt* | [dictionary-literal-item](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_dictionary-literal-item) **,** [dictionary-literal-items](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_dictionary-literal-items)

*dictionary-literal-item* → [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **:** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression)

*playground-literal* → **#colorLiteral** **(** **red** **:** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **,** **green** **:** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **,** **blue** **:** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **,** **alpha** **:** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **)**

*playground-literal* → **#fileLiteral** **(** **resourceName** **:** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **)**

*playground-literal* → **#imageLiteral** **(** **resourceName** **:** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **)**

GRAMMAR OF A SELF EXPRESSION

*self-expression* → **self** | [self-method-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_self-method-expression) | [self-subscript-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_self-subscript-expression) | [self-initializer-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_self-initializer-expression)

*self-method-expression* → **self** **.** [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*self-subscript-expression* → **self** **[** [function-call-argument-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_function-call-argument-list) **]**

*self-initializer-expression* → **self** **.** **init**

GRAMMAR OF A SUPERCLASS EXPRESSION

*superclass-expression* → [superclass-method-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_superclass-method-expression) | [superclass-subscript-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_superclass-subscript-expression) | [superclass-initializer-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_superclass-initializer-expression)

*superclass-method-expression* → **super** **.** [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*superclass-subscript-expression* → **super** **[** [function-call-argument-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_function-call-argument-list) **]**

*superclass-initializer-expression* → **super** **.** **init**

GRAMMAR OF A CLOSURE EXPRESSION

*closure-expression* → **{** [closure-signature](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_closure-signature) *opt* [statements](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statements) *opt* **}**

*closure-signature* → [capture-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_capture-list) *opt* [closure-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_closure-parameter-clause) **throws***opt* [function-result](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_function-result) *opt* **in**

*closure-signature* → [capture-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_capture-list) **in**

*closure-parameter-clause* → **(** **)** | **(** [closure-parameter-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_closure-parameter-list) **)** | [identifier-list](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier-list)

*closure-parameter-list* → [closure-parameter](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_closure-parameter) | [closure-parameter](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_closure-parameter) **,** [closure-parameter-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_closure-parameter-list)

*closure-parameter* → [closure-parameter-name](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_closure-parameter-name) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) *opt*

*closure-parameter* → [closure-parameter-name](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_closure-parameter-name) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) **...**

*closure-parameter-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*capture-list* → **[** [capture-list-items](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_capture-list-items) **]**

*capture-list-items* → [capture-list-item](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_capture-list-item) | [capture-list-item](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_capture-list-item) **,** [capture-list-items](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_capture-list-items)

*capture-list-item* → [capture-specifier](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_capture-specifier) *opt* [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*capture-list-item* → [capture-specifier](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_capture-specifier) *opt* [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) **=** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression)

*capture-list-item* → [capture-specifier](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_capture-specifier) *opt* [self-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_self-expression)

*capture-specifier* → **weak** | **unowned** | **unowned(safe)** | **unowned(unsafe)**

GRAMMAR OF A IMPLICIT MEMBER EXPRESSION

*implicit-member-expression* → **.** [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

GRAMMAR OF A PARENTHESIZED EXPRESSION

*parenthesized-expression* → **(** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **)**

GRAMMAR OF A TUPLE EXPRESSION

*tuple-expression* → **(** **)** | **(** [tuple-element](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_tuple-element) **,** [tuple-element-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_tuple-element-list) **)**

*tuple-element-list* → [tuple-element](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_tuple-element) | [tuple-element](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_tuple-element) **,** [tuple-element-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_tuple-element-list)

*tuple-element* → [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) | [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) **:** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression)

GRAMMAR OF A WILDCARD EXPRESSION

*wildcard-expression* → **\_**

GRAMMAR OF A KEY-PATH EXPRESSION

*key-path-expression* → **\** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type) *opt* **.** [key-path-components](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_key-path-components)

*key-path-components* → [key-path-component](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_key-path-component) | [key-path-component](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_key-path-component) **.** [key-path-components](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_key-path-components)

*key-path-component* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) [key-path-postfixes](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_key-path-postfixes) *opt* | [key-path-postfixes](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_key-path-postfixes)

*key-path-postfixes* → [key-path-postfix](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_key-path-postfix) [key-path-postfixes](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_key-path-postfixes) *opt*

*key-path-postfix* → **?** | **!** | **self** | **[** [function-call-argument-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_function-call-argument-list) **]**

GRAMMAR OF A SELECTOR EXPRESSION

*selector-expression* → **#selector** **(** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **)**

*selector-expression* → **#selector** **(** **getter:** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **)**

*selector-expression* → **#selector** **(** **setter:** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **)**

GRAMMAR OF A KEY-PATH STRING EXPRESSION

*key-path-string-expression* → **#keyPath** **(** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **)**

GRAMMAR OF A POSTFIX EXPRESSION

*postfix-expression* → [primary-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_primary-expression)

*postfix-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) [postfix-operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_postfix-operator)

*postfix-expression* → [function-call-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_function-call-expression)

*postfix-expression* → [initializer-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_initializer-expression)

*postfix-expression* → [explicit-member-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_explicit-member-expression)

*postfix-expression* → [postfix-self-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-self-expression)

*postfix-expression* → [subscript-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_subscript-expression)

*postfix-expression* → [forced-value-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_forced-value-expression)

*postfix-expression* → [optional-chaining-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_optional-chaining-expression)

GRAMMAR OF A FUNCTION CALL EXPRESSION

*function-call-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) [function-call-argument-clause](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_function-call-argument-clause)

*function-call-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) [function-call-argument-clause](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_function-call-argument-clause) *opt* [trailing-closures](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_trailing-closures)

*function-call-argument-clause* → **(** **)** | **(** [function-call-argument-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_function-call-argument-list) **)**

*function-call-argument-list* → [function-call-argument](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_function-call-argument) | [function-call-argument](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_function-call-argument) **,** [function-call-argument-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_function-call-argument-list)

*function-call-argument* → [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) | [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) **:** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression)

*function-call-argument* → [operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator) | [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) **:** [operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator)

*trailing-closures* → [closure-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_closure-expression) [labeled-trailing-closures](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_labeled-trailing-closures) *opt*

*labeled-trailing-closures* → [labeled-trailing-closure](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_labeled-trailing-closure) [labeled-trailing-closures](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_labeled-trailing-closures) *opt*

*labeled-trailing-closure* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) **:** [closure-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_closure-expression)

GRAMMAR OF AN INITIALIZER EXPRESSION

*initializer-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) **.** **init**

*initializer-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) **.** **init** **(** [argument-names](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_argument-names) **)**

GRAMMAR OF AN EXPLICIT MEMBER EXPRESSION

*explicit-member-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) **.** [decimal-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digits)

*explicit-member-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) **.** [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) [generic-argument-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-argument-clause) *opt*

*explicit-member-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) **.** [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) **(** [argument-names](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_argument-names) **)**

*argument-names* → [argument-name](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_argument-name) [argument-names](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_argument-names) *opt*

*argument-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) **:**

GRAMMAR OF A POSTFIX SELF EXPRESSION

*postfix-self-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) **.** **self**

GRAMMAR OF A SUBSCRIPT EXPRESSION

*subscript-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) **[** [function-call-argument-list](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_function-call-argument-list) **]**

GRAMMAR OF A FORCED-VALUE EXPRESSION

*forced-value-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) **!**

GRAMMAR OF AN OPTIONAL-CHAINING EXPRESSION

*optional-chaining-expression* → [postfix-expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_postfix-expression) **?**

Statements

GRAMMAR OF A STATEMENT

*statement* → [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **;***opt*

*statement* → [declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration) **;***opt*

*statement* → [loop-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_loop-statement) **;***opt*

*statement* → [branch-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_branch-statement) **;***opt*

*statement* → [labeled-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_labeled-statement) **;***opt*

*statement* → [control-transfer-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_control-transfer-statement) **;***opt*

*statement* → [defer-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_defer-statement) **;***opt*

*statement* → [do-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_do-statement) **;***opt*

*statement* → [compiler-control-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compiler-control-statement)

*statements* → [statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statement) [statements](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statements) *opt*

GRAMMAR OF A LOOP STATEMENT

*loop-statement* → [for-in-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_for-in-statement)

*loop-statement* → [while-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_while-statement)

*loop-statement* → [repeat-while-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_repeat-while-statement)

GRAMMAR OF A FOR-IN STATEMENT

*for-in-statement* → **for** **case***opt* [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern) **in** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) [where-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_where-clause) *opt* [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

GRAMMAR OF A WHILE STATEMENT

*while-statement* → **while** [condition-list](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_condition-list) [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

*condition-list* → [condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_condition) | [condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_condition) **,** [condition-list](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_condition-list)

*condition* → [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) | [availability-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_availability-condition) | [case-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_case-condition) | [optional-binding-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_optional-binding-condition)

*case-condition* → **case** [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern) [initializer](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer)

*optional-binding-condition* → **let** [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern) [initializer](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer) | **var** [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern) [initializer](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer)

GRAMMAR OF A REPEAT-WHILE STATEMENT

*repeat-while-statement* → **repeat** [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block) **while** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression)

GRAMMAR OF A BRANCH STATEMENT

*branch-statement* → [if-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_if-statement)

*branch-statement* → [guard-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_guard-statement)

*branch-statement* → [switch-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-statement)

GRAMMAR OF AN IF STATEMENT

*if-statement* → **if** [condition-list](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_condition-list) [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block) [else-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_else-clause) *opt*

*else-clause* → **else** [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block) | **else** [if-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_if-statement)

GRAMMAR OF A GUARD STATEMENT

*guard-statement* → **guard** [condition-list](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_condition-list) **else** [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

GRAMMAR OF A SWITCH STATEMENT

*switch-statement* → **switch** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) **{** [switch-cases](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-cases) *opt* **}**

*switch-cases* → [switch-case](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-case) [switch-cases](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-cases) *opt*

*switch-case* → [case-label](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_case-label) [statements](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statements)

*switch-case* → [default-label](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_default-label) [statements](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statements)

*switch-case* → [conditional-switch-case](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_conditional-switch-case)

*case-label* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* **case** [case-item-list](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_case-item-list) **:**

*case-item-list* → [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern) [where-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_where-clause) *opt* | [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern) [where-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_where-clause) *opt* **,** [case-item-list](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_case-item-list)

*default-label* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* **default** **:**

*where-clause* → **where** [where-expression](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_where-expression)

*where-expression* → [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression)

*conditional-switch-case* → [switch-if-directive-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-if-directive-clause) [switch-elseif-directive-clauses](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-elseif-directive-clauses) *opt* [switch-else-directive-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-else-directive-clause) *opt* [endif-directive](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_endif-directive)

*switch-if-directive-clause* → [if-directive](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_if-directive) [compilation-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compilation-condition) [switch-cases](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-cases) *opt*

*switch-elseif-directive-clauses* → [elseif-directive-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_elseif-directive-clause) [switch-elseif-directive-clauses](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-elseif-directive-clauses) *opt*

*switch-elseif-directive-clause* → [elseif-directive](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_elseif-directive) [compilation-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compilation-condition) [switch-cases](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-cases) *opt*

*switch-else-directive-clause* → [else-directive](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_else-directive) [switch-cases](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-cases) *opt*

GRAMMAR OF A LABELED STATEMENT

*labeled-statement* → [statement-label](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statement-label) [loop-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_loop-statement)

*labeled-statement* → [statement-label](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statement-label) [if-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_if-statement)

*labeled-statement* → [statement-label](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statement-label) [switch-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_switch-statement)

*labeled-statement* → [statement-label](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statement-label) [do-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_do-statement)

*statement-label* → [label-name](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_label-name) **:**

*label-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

GRAMMAR OF A CONTROL TRANSFER STATEMENT

*control-transfer-statement* → [break-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_break-statement)

*control-transfer-statement* → [continue-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_continue-statement)

*control-transfer-statement* → [fallthrough-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_fallthrough-statement)

*control-transfer-statement* → [return-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_return-statement)

*control-transfer-statement* → [throw-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_throw-statement)

GRAMMAR OF A BREAK STATEMENT

*break-statement* → **break** [label-name](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_label-name) *opt*

GRAMMAR OF A CONTINUE STATEMENT

*continue-statement* → **continue** [label-name](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_label-name) *opt*

GRAMMAR OF A FALLTHROUGH STATEMENT

*fallthrough-statement* → **fallthrough**

GRAMMAR OF A RETURN STATEMENT

*return-statement* → **return** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression) *opt*

GRAMMAR OF A THROW STATEMENT

*throw-statement* → **throw** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression)

GRAMMAR OF A DEFER STATEMENT

*defer-statement* → **defer** [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

GRAMMAR OF A DO STATEMENT

*do-statement* → **do** [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block) [catch-clauses](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_catch-clauses) *opt*

*catch-clauses* → [catch-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_catch-clause) [catch-clauses](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_catch-clauses) *opt*

*catch-clause* → **catch** [catch-pattern-list](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_catch-pattern-list) *opt* [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

*catch-pattern-list* → [catch-pattern](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_catch-pattern) | [catch-pattern](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_catch-pattern) **,** [catch-pattern-list](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_catch-pattern-list)

*catch-pattern* → [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern) [where-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_where-clause) *opt*

GRAMMAR OF A COMPILER CONTROL STATEMENT

*compiler-control-statement* → [conditional-compilation-block](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_conditional-compilation-block)

*compiler-control-statement* → [line-control-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_line-control-statement)

*compiler-control-statement* → [diagnostic-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_diagnostic-statement)

GRAMMAR OF A CONDITIONAL COMPILATION BLOCK

*conditional-compilation-block* → [if-directive-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_if-directive-clause) [elseif-directive-clauses](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_elseif-directive-clauses) *opt* [else-directive-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_else-directive-clause) *opt* [endif-directive](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_endif-directive)

*if-directive-clause* → [if-directive](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_if-directive) [compilation-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compilation-condition) [statements](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statements) *opt*

*elseif-directive-clauses* → [elseif-directive-clause](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_elseif-directive-clause) [elseif-directive-clauses](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_elseif-directive-clauses) *opt*

*elseif-directive-clause* → [elseif-directive](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_elseif-directive) [compilation-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compilation-condition) [statements](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statements) *opt*

*else-directive-clause* → [else-directive](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_else-directive) [statements](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statements) *opt*

*if-directive* → **#if**

*elseif-directive* → **#elseif**

*else-directive* → **#else**

*endif-directive* → **#endif**

*compilation-condition* → [platform-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_platform-condition)

*compilation-condition* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*compilation-condition* → [boolean-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_boolean-literal)

*compilation-condition* → **(** [compilation-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compilation-condition) **)**

*compilation-condition* → **!** [compilation-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compilation-condition)

*compilation-condition* → [compilation-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compilation-condition) **&&** [compilation-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compilation-condition)

*compilation-condition* → [compilation-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compilation-condition) **||** [compilation-condition](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compilation-condition)

*platform-condition* → **os** **(** [operating-system](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_operating-system) **)**

*platform-condition* → **arch** **(** [architecture](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_architecture) **)**

*platform-condition* → **swift** **(** **>=** [swift-version](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_swift-version) **)** | **swift** **(** **<** [swift-version](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_swift-version) **)**

*platform-condition* → **compiler** **(** **>=** [swift-version](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_swift-version) **)** | **compiler** **(** **<** [swift-version](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_swift-version) **)**

*platform-condition* → **canImport** **(** [module-name](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_module-name) **)**

*platform-condition* → **targetEnvironment** **(** [environment](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_environment) **)**

*operating-system* → **macOS** | **iOS** | **watchOS** | **tvOS** | **Linux** | **Windows**

*architecture* → **i386** | **x86\_64** | **arm** | **arm64**

*swift-version* → [decimal-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digits) [swift-version-continuation](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_swift-version-continuation) *opt*

*swift-version-continuation* → **.** [decimal-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digits) [swift-version-continuation](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_swift-version-continuation) *opt*

*module-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*environment* → **simulator** | **macCatalyst**

GRAMMAR OF A LINE CONTROL STATEMENT

*line-control-statement* → **#sourceLocation** **(** **file:** [file-path](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_file-path) **,** **line:** [line-number](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_line-number) **)**

*line-control-statement* → **#sourceLocation** **(** **)**

*line-number* → A decimal integer greater than zero

*file-path* → [static-string-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_static-string-literal)

GRAMMAR OF A COMPILE-TIME DIAGNOSTIC STATEMENT

*diagnostic-statement* → **#error** **(** [diagnostic-message](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_diagnostic-message) **)**

*diagnostic-statement* → **#warning** **(** [diagnostic-message](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_diagnostic-message) **)**

*diagnostic-message* → [static-string-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_static-string-literal)

GRAMMAR OF AN AVAILABILITY CONDITION

*availability-condition* → **#available** **(** [availability-arguments](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_availability-arguments) **)**

*availability-arguments* → [availability-argument](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_availability-argument) | [availability-argument](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_availability-argument) **,** [availability-arguments](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_availability-arguments)

*availability-argument* → [platform-name](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_platform-name) [platform-version](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_platform-version)

*availability-argument* → **\***

*platform-name* → **iOS** | **iOSApplicationExtension**

*platform-name* → **macOS** | **macOSApplicationExtension**

*platform-name* → **macCatalyst** | **macCatalystApplicationExtension**

*platform-name* → **watchOS**

*platform-name* → **tvOS**

*platform-version* → [decimal-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digits)

*platform-version* → [decimal-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digits) **.** [decimal-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digits)

*platform-version* → [decimal-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digits) **.** [decimal-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digits) **.** [decimal-digits](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_decimal-digits)

Declarations

GRAMMAR OF A DECLARATION

*declaration* → [import-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_import-declaration)

*declaration* → [constant-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_constant-declaration)

*declaration* → [variable-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-declaration)

*declaration* → [typealias-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_typealias-declaration)

*declaration* → [function-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_function-declaration)

*declaration* → [enum-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_enum-declaration)

*declaration* → [struct-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_struct-declaration)

*declaration* → [class-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_class-declaration)

*declaration* → [protocol-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-declaration)

*declaration* → [initializer-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer-declaration)

*declaration* → [deinitializer-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_deinitializer-declaration)

*declaration* → [extension-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_extension-declaration)

*declaration* → [subscript-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_subscript-declaration)

*declaration* → [operator-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_operator-declaration)

*declaration* → [precedence-group-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-declaration)

*declarations* → [declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration) [declarations](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declarations) *opt*

GRAMMAR OF A TOP-LEVEL DECLARATION

*top-level-declaration* → [statements](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statements) *opt*

GRAMMAR OF A CODE BLOCK

*code-block* → **{** [statements](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_statements) *opt* **}**

GRAMMAR OF AN IMPORT DECLARATION

*import-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* **import** [import-kind](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_import-kind) *opt* [import-path](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_import-path)

*import-kind* → **typealias** | **struct** | **class** | **enum** | **protocol** | **let** | **var** | **func**

*import-path* → [import-path-identifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_import-path-identifier) | [import-path-identifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_import-path-identifier) **.** [import-path](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_import-path)

*import-path-identifier* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) | [operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator)

GRAMMAR OF A CONSTANT DECLARATION

*constant-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [declaration-modifiers](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration-modifiers) *opt* **let** [pattern-initializer-list](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_pattern-initializer-list)

*pattern-initializer-list* → [pattern-initializer](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_pattern-initializer) | [pattern-initializer](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_pattern-initializer) **,** [pattern-initializer-list](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_pattern-initializer-list)

*pattern-initializer* → [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern) [initializer](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer) *opt*

*initializer* → **=** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression)

GRAMMAR OF A VARIABLE DECLARATION

*variable-declaration* → [variable-declaration-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-declaration-head) [pattern-initializer-list](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_pattern-initializer-list)

*variable-declaration* → [variable-declaration-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-declaration-head) [variable-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-name) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

*variable-declaration* → [variable-declaration-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-declaration-head) [variable-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-name) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) [getter-setter-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_getter-setter-block)

*variable-declaration* → [variable-declaration-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-declaration-head) [variable-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-name) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) [getter-setter-keyword-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_getter-setter-keyword-block)

*variable-declaration* → [variable-declaration-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-declaration-head) [variable-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-name) [initializer](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer) [willSet-didSet-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_willSet-didSet-block)

*variable-declaration* → [variable-declaration-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-declaration-head) [variable-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-name) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) [initializer](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer) *opt* [willSet-didSet-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_willSet-didSet-block)

*variable-declaration-head* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [declaration-modifiers](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration-modifiers) *opt* **var**

*variable-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*getter-setter-block* → [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

*getter-setter-block* → **{** [getter-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_getter-clause) [setter-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_setter-clause) *opt* **}**

*getter-setter-block* → **{** [setter-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_setter-clause) [getter-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_getter-clause) **}**

*getter-clause* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [mutation-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_mutation-modifier) *opt* **get** [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

*setter-clause* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [mutation-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_mutation-modifier) *opt* **set** [setter-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_setter-name) *opt* [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

*setter-name* → **(** [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) **)**

*getter-setter-keyword-block* → **{** [getter-keyword-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_getter-keyword-clause) [setter-keyword-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_setter-keyword-clause) *opt* **}**

*getter-setter-keyword-block* → **{** [setter-keyword-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_setter-keyword-clause) [getter-keyword-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_getter-keyword-clause) **}**

*getter-keyword-clause* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [mutation-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_mutation-modifier) *opt* **get**

*setter-keyword-clause* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [mutation-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_mutation-modifier) *opt* **set**

*willSet-didSet-block* → **{** [willSet-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_willSet-clause) [didSet-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_didSet-clause) *opt* **}**

*willSet-didSet-block* → **{** [didSet-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_didSet-clause) [willSet-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_willSet-clause) *opt* **}**

*willSet-clause* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* **willSet** [setter-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_setter-name) *opt* [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

*didSet-clause* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* **didSet** [setter-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_setter-name) *opt* [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

GRAMMAR OF A TYPE ALIAS DECLARATION

*typealias-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [access-level-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_access-level-modifier) *opt* **typealias** [typealias-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_typealias-name) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [typealias-assignment](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_typealias-assignment)

*typealias-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*typealias-assignment* → **=** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

GRAMMAR OF A FUNCTION DECLARATION

*function-declaration* → [function-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_function-head) [function-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_function-name) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [function-signature](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_function-signature) [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [function-body](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_function-body) *opt*

*function-head* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [declaration-modifiers](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration-modifiers) *opt* **func**

*function-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) | [operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator)

*function-signature* → [parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_parameter-clause) **throws***opt* [function-result](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_function-result) *opt*

*function-signature* → [parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_parameter-clause) **rethrows** [function-result](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_function-result) *opt*

*function-result* → **->** [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

*function-body* → [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

*parameter-clause* → **(** **)** | **(** [parameter-list](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_parameter-list) **)**

*parameter-list* → [parameter](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_parameter) | [parameter](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_parameter) **,** [parameter-list](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_parameter-list)

*parameter* → [external-parameter-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_external-parameter-name) *opt* [local-parameter-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_local-parameter-name) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) [default-argument-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_default-argument-clause) *opt*

*parameter* → [external-parameter-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_external-parameter-name) *opt* [local-parameter-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_local-parameter-name) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation)

*parameter* → [external-parameter-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_external-parameter-name) *opt* [local-parameter-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_local-parameter-name) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) **...**

*external-parameter-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*local-parameter-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*default-argument-clause* → **=** [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression)

GRAMMAR OF AN ENUMERATION DECLARATION

*enum-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [access-level-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_access-level-modifier) *opt* [union-style-enum](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_union-style-enum)

*enum-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [access-level-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_access-level-modifier) *opt* [raw-value-style-enum](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_raw-value-style-enum)

*union-style-enum* → **indirect***opt* **enum** [enum-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_enum-name) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [type-inheritance-clause](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-inheritance-clause) *opt* [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* **{** [union-style-enum-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_union-style-enum-members) *opt* **}**

*union-style-enum-members* → [union-style-enum-member](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_union-style-enum-member) [union-style-enum-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_union-style-enum-members) *opt*

*union-style-enum-member* → [declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration) | [union-style-enum-case-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_union-style-enum-case-clause) | [compiler-control-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compiler-control-statement)

*union-style-enum-case-clause* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* **indirect***opt* **case** [union-style-enum-case-list](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_union-style-enum-case-list)

*union-style-enum-case-list* → [union-style-enum-case](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_union-style-enum-case) | [union-style-enum-case](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_union-style-enum-case) **,** [union-style-enum-case-list](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_union-style-enum-case-list)

*union-style-enum-case* → [enum-case-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_enum-case-name) [tuple-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_tuple-type) *opt*

*enum-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*enum-case-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*raw-value-style-enum* → **enum** [enum-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_enum-name) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [type-inheritance-clause](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-inheritance-clause) [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* **{** [raw-value-style-enum-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_raw-value-style-enum-members) **}**

*raw-value-style-enum-members* → [raw-value-style-enum-member](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_raw-value-style-enum-member) [raw-value-style-enum-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_raw-value-style-enum-members) *opt*

*raw-value-style-enum-member* → [declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration) | [raw-value-style-enum-case-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_raw-value-style-enum-case-clause) | [compiler-control-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compiler-control-statement)

*raw-value-style-enum-case-clause* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* **case** [raw-value-style-enum-case-list](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_raw-value-style-enum-case-list)

*raw-value-style-enum-case-list* → [raw-value-style-enum-case](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_raw-value-style-enum-case) | [raw-value-style-enum-case](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_raw-value-style-enum-case) **,** [raw-value-style-enum-case-list](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_raw-value-style-enum-case-list)

*raw-value-style-enum-case* → [enum-case-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_enum-case-name) [raw-value-assignment](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_raw-value-assignment) *opt*

*raw-value-assignment* → **=** [raw-value-literal](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_raw-value-literal)

*raw-value-literal* → [numeric-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_numeric-literal) | [static-string-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_static-string-literal) | [boolean-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_boolean-literal)

GRAMMAR OF A STRUCTURE DECLARATION

*struct-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [access-level-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_access-level-modifier) *opt* **struct** [struct-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_struct-name) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [type-inheritance-clause](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-inheritance-clause) *opt* [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [struct-body](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_struct-body)

*struct-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*struct-body* → **{** [struct-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_struct-members) *opt* **}**

*struct-members* → [struct-member](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_struct-member) [struct-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_struct-members) *opt*

*struct-member* → [declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration) | [compiler-control-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compiler-control-statement)

GRAMMAR OF A CLASS DECLARATION

*class-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [access-level-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_access-level-modifier) *opt* **final***opt* **class** [class-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_class-name) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [type-inheritance-clause](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-inheritance-clause) *opt* [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [class-body](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_class-body)

*class-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* **final** [access-level-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_access-level-modifier) *opt* **class** [class-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_class-name) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [type-inheritance-clause](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-inheritance-clause) *opt* [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [class-body](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_class-body)

*class-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*class-body* → **{** [class-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_class-members) *opt* **}**

*class-members* → [class-member](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_class-member) [class-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_class-members) *opt*

*class-member* → [declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration) | [compiler-control-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compiler-control-statement)

GRAMMAR OF A PROTOCOL DECLARATION

*protocol-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [access-level-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_access-level-modifier) *opt* **protocol** [protocol-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-name) [type-inheritance-clause](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-inheritance-clause) *opt* [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [protocol-body](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-body)

*protocol-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*protocol-body* → **{** [protocol-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-members) *opt* **}**

*protocol-members* → [protocol-member](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-member) [protocol-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-members) *opt*

*protocol-member* → [protocol-member-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-member-declaration) | [compiler-control-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compiler-control-statement)

*protocol-member-declaration* → [protocol-property-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-property-declaration)

*protocol-member-declaration* → [protocol-method-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-method-declaration)

*protocol-member-declaration* → [protocol-initializer-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-initializer-declaration)

*protocol-member-declaration* → [protocol-subscript-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-subscript-declaration)

*protocol-member-declaration* → [protocol-associated-type-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_protocol-associated-type-declaration)

*protocol-member-declaration* → [typealias-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_typealias-declaration)

GRAMMAR OF A PROTOCOL PROPERTY DECLARATION

*protocol-property-declaration* → [variable-declaration-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-declaration-head) [variable-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_variable-name) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) [getter-setter-keyword-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_getter-setter-keyword-block)

GRAMMAR OF A PROTOCOL METHOD DECLARATION

*protocol-method-declaration* → [function-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_function-head) [function-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_function-name) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [function-signature](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_function-signature) [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt*

GRAMMAR OF A PROTOCOL INITIALIZER DECLARATION

*protocol-initializer-declaration* → [initializer-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer-head) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_parameter-clause) **throws***opt* [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt*

*protocol-initializer-declaration* → [initializer-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer-head) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_parameter-clause) **rethrows** [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt*

GRAMMAR OF A PROTOCOL SUBSCRIPT DECLARATION

*protocol-subscript-declaration* → [subscript-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_subscript-head) [subscript-result](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_subscript-result) [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [getter-setter-keyword-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_getter-setter-keyword-block)

GRAMMAR OF A PROTOCOL ASSOCIATED TYPE DECLARATION

*protocol-associated-type-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [access-level-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_access-level-modifier) *opt* **associatedtype** [typealias-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_typealias-name) [type-inheritance-clause](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-inheritance-clause) *opt* [typealias-assignment](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_typealias-assignment) *opt* [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt*

GRAMMAR OF AN INITIALIZER DECLARATION

*initializer-declaration* → [initializer-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer-head) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_parameter-clause) **throws***opt* [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [initializer-body](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer-body)

*initializer-declaration* → [initializer-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer-head) [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_parameter-clause) **rethrows** [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [initializer-body](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_initializer-body)

*initializer-head* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [declaration-modifiers](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration-modifiers) *opt* **init**

*initializer-head* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [declaration-modifiers](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration-modifiers) *opt* **init** **?**

*initializer-head* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [declaration-modifiers](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration-modifiers) *opt* **init** **!**

*initializer-body* → [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

GRAMMAR OF A DEINITIALIZER DECLARATION

*deinitializer-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* **deinit** [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

GRAMMAR OF AN EXTENSION DECLARATION

*extension-declaration* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [access-level-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_access-level-modifier) *opt* **extension** [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier) [type-inheritance-clause](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-inheritance-clause) *opt* [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [extension-body](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_extension-body)

*extension-body* → **{** [extension-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_extension-members) *opt* **}**

*extension-members* → [extension-member](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_extension-member) [extension-members](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_extension-members) *opt*

*extension-member* → [declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration) | [compiler-control-statement](https://docs.swift.org/swift-book/ReferenceManual/Statements.html#grammar_compiler-control-statement)

GRAMMAR OF A SUBSCRIPT DECLARATION

*subscript-declaration* → [subscript-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_subscript-head) [subscript-result](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_subscript-result) [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [code-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_code-block)

*subscript-declaration* → [subscript-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_subscript-head) [subscript-result](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_subscript-result) [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [getter-setter-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_getter-setter-block)

*subscript-declaration* → [subscript-head](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_subscript-head) [subscript-result](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_subscript-result) [generic-where-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-where-clause) *opt* [getter-setter-keyword-block](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_getter-setter-keyword-block)

*subscript-head* → [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [declaration-modifiers](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration-modifiers) *opt* **subscript** [generic-parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-clause) *opt* [parameter-clause](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_parameter-clause)

*subscript-result* → **->** [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt* [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

GRAMMAR OF AN OPERATOR DECLARATION

*operator-declaration* → [prefix-operator-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_prefix-operator-declaration) | [postfix-operator-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_postfix-operator-declaration) | [infix-operator-declaration](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_infix-operator-declaration)

*prefix-operator-declaration* → **prefix** **operator** [operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator)

*postfix-operator-declaration* → **postfix** **operator** [operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator)

*infix-operator-declaration* → **infix** **operator** [operator](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_operator) [infix-operator-group](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_infix-operator-group) *opt*

*infix-operator-group* → **:** [precedence-group-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-name)

GRAMMAR OF A PRECEDENCE GROUP DECLARATION

*precedence-group-declaration* → **precedencegroup** [precedence-group-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-name) **{** [precedence-group-attributes](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-attributes) *opt* **}**

*precedence-group-attributes* → [precedence-group-attribute](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-attribute) [precedence-group-attributes](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-attributes) *opt*

*precedence-group-attribute* → [precedence-group-relation](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-relation)

*precedence-group-attribute* → [precedence-group-assignment](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-assignment)

*precedence-group-attribute* → [precedence-group-associativity](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-associativity)

*precedence-group-relation* → **higherThan** **:** [precedence-group-names](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-names)

*precedence-group-relation* → **lowerThan** **:** [precedence-group-names](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-names)

*precedence-group-assignment* → **assignment** **:** [boolean-literal](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_boolean-literal)

*precedence-group-associativity* → **associativity** **:** **left**

*precedence-group-associativity* → **associativity** **:** **right**

*precedence-group-associativity* → **associativity** **:** **none**

*precedence-group-names* → [precedence-group-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-name) | [precedence-group-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-name) **,** [precedence-group-names](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_precedence-group-names)

*precedence-group-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

GRAMMAR OF A DECLARATION MODIFIER

*declaration-modifier* → **class** | **convenience** | **dynamic** | **final** | **infix** | **lazy** | **optional** | **override** | **postfix** | **prefix** | **required** | **static** | **unowned** | **unowned** **(** **safe** **)** | **unowned** **(** **unsafe** **)** | **weak**

*declaration-modifier* → [access-level-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_access-level-modifier)

*declaration-modifier* → [mutation-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_mutation-modifier)

*declaration-modifiers* → [declaration-modifier](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration-modifier) [declaration-modifiers](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_declaration-modifiers) *opt*

*access-level-modifier* → **private** | **private** **(** **set** **)**

*access-level-modifier* → **fileprivate** | **fileprivate** **(** **set** **)**

*access-level-modifier* → **internal** | **internal** **(** **set** **)**

*access-level-modifier* → **public** | **public** **(** **set** **)**

*access-level-modifier* → **open** | **open** **(** **set** **)**

*mutation-modifier* → **mutating** | **nonmutating**

Attributes

GRAMMAR OF AN ATTRIBUTE

*attribute* → **@** [attribute-name](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attribute-name) [attribute-argument-clause](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attribute-argument-clause) *opt*

*attribute-name* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

*attribute-argument-clause* → **(** [balanced-tokens](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_balanced-tokens) *opt* **)**

*attributes* → [attribute](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attribute) [attributes](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_attributes) *opt*

*balanced-tokens* → [balanced-token](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_balanced-token) [balanced-tokens](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_balanced-tokens) *opt*

*balanced-token* → **(** [balanced-tokens](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_balanced-tokens) *opt* **)**

*balanced-token* → **[** [balanced-tokens](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_balanced-tokens) *opt* **]**

*balanced-token* → **{** [balanced-tokens](https://docs.swift.org/swift-book/ReferenceManual/Attributes.html#grammar_balanced-tokens) *opt* **}**

*balanced-token* → Any identifier, keyword, literal, or operator

*balanced-token* → Any punctuation except **(**, **)**, **[**, **]**, **{**, or **}**

Patterns

GRAMMAR OF A PATTERN

*pattern* → [wildcard-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_wildcard-pattern) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) *opt*

*pattern* → [identifier-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_identifier-pattern) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) *opt*

*pattern* → [value-binding-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_value-binding-pattern)

*pattern* → [tuple-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_tuple-pattern) [type-annotation](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-annotation) *opt*

*pattern* → [enum-case-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_enum-case-pattern)

*pattern* → [optional-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_optional-pattern)

*pattern* → [type-casting-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_type-casting-pattern)

*pattern* → [expression-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_expression-pattern)

GRAMMAR OF A WILDCARD PATTERN

*wildcard-pattern* → **\_**

GRAMMAR OF AN IDENTIFIER PATTERN

*identifier-pattern* → [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier)

GRAMMAR OF A VALUE-BINDING PATTERN

*value-binding-pattern* → **var** [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern) | **let** [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern)

GRAMMAR OF A TUPLE PATTERN

*tuple-pattern* → **(** [tuple-pattern-element-list](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_tuple-pattern-element-list) *opt* **)**

*tuple-pattern-element-list* → [tuple-pattern-element](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_tuple-pattern-element) | [tuple-pattern-element](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_tuple-pattern-element) **,** [tuple-pattern-element-list](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_tuple-pattern-element-list)

*tuple-pattern-element* → [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern) | [identifier](https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#grammar_identifier) **:** [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern)

GRAMMAR OF AN ENUMERATION CASE PATTERN

*enum-case-pattern* → [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier) *opt* **.** [enum-case-name](https://docs.swift.org/swift-book/ReferenceManual/Declarations.html#grammar_enum-case-name) [tuple-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_tuple-pattern) *opt*

GRAMMAR OF AN OPTIONAL PATTERN

*optional-pattern* → [identifier-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_identifier-pattern) **?**

GRAMMAR OF A TYPE CASTING PATTERN

*type-casting-pattern* → [is-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_is-pattern) | [as-pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_as-pattern)

*is-pattern* → **is** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

*as-pattern* → [pattern](https://docs.swift.org/swift-book/ReferenceManual/Patterns.html#grammar_pattern) **as** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

GRAMMAR OF AN EXPRESSION PATTERN

*expression-pattern* → [expression](https://docs.swift.org/swift-book/ReferenceManual/Expressions.html#grammar_expression)

Generic Parameters and Arguments

GRAMMAR OF A GENERIC PARAMETER CLAUSE

*generic-parameter-clause* → **<** [generic-parameter-list](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-list) **>**

*generic-parameter-list* → [generic-parameter](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter) | [generic-parameter](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter) **,** [generic-parameter-list](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-parameter-list)

*generic-parameter* → [type-name](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-name)

*generic-parameter* → [type-name](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-name) **:** [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier)

*generic-parameter* → [type-name](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-name) **:** [protocol-composition-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_protocol-composition-type)

*generic-where-clause* → **where** [requirement-list](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_requirement-list)

*requirement-list* → [requirement](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_requirement) | [requirement](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_requirement) **,** [requirement-list](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_requirement-list)

*requirement* → [conformance-requirement](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_conformance-requirement) | [same-type-requirement](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_same-type-requirement)

*conformance-requirement* → [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier) **:** [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier)

*conformance-requirement* → [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier) **:** [protocol-composition-type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_protocol-composition-type)

*same-type-requirement* → [type-identifier](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type-identifier) **==** [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)

GRAMMAR OF A GENERIC ARGUMENT CLAUSE

*generic-argument-clause* → **<** [generic-argument-list](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-argument-list) **>**

*generic-argument-list* → [generic-argument](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-argument) | [generic-argument](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-argument) **,** [generic-argument-list](https://docs.swift.org/swift-book/ReferenceManual/GenericParametersAndArguments.html#grammar_generic-argument-list)

*generic-argument* → [type](https://docs.swift.org/swift-book/ReferenceManual/Types.html#grammar_type)