* 1. Syntactic grammar
     1. Basic concepts

namespace-name:  
namespace-or-type-name

type-name:  
namespace-or-type-name

namespace-or-type-name:  
identifier type-argument-listopt  
namespace-or-type-name . identifier type-argument-listoptqualified-alias-member

* + 1. Types

type:  
value-type  
reference-type   
type-parameter

value-type:  
struct-type  
enum-type

struct-type:  
type-name  
simple-type   
nullable-type

simple-type:  
numeric-type  
bool

numeric-type:  
integral-type  
floating-point-type  
decimal

integral-type:  
sbyte  
byte  
short  
ushort  
int  
uint  
long  
ulong  
char

floating-point-type:  
float  
double

nullable-type:  
non-nullable-value-type ?

non-nullable-value-type:  
type

enum-type:  
type-name

reference-type:  
class-type  
interface-type  
array-type  
delegate-type

class-type:  
type-name  
object  
string

interface-type:  
type-name

array-type:  
non-array-type rank-specifiers

non-array-type:  
type

rank-specifiers:  
rank-specifier  
rank-specifiers rank-specifier

rank-specifier:  
[ dim-separatorsopt ]

dim-separators:  
,  
dim-separators ,

delegate-type:  
type-name

type-argument-list:  
< type-arguments >

type-arguments:  
type-argument  
type-arguments , type-argument

type-argument:  
type

type-parameter:  
identifier

* + 1. Variables

variable-reference:  
expression

* + 1. Expressions

argument-list:  
argument  
argument-list , argument

argument:  
expression  
ref variable-reference  
out variable-reference

primary-expression:   
primary-no-array-creation-expression  
array-creation-expression

primary-no-array-creation-expression:  
literal  
simple-name  
parenthesized-expression  
member-access  
invocation-expression  
element-access  
this-access  
base-access  
post-increment-expression  
post-decrement-expression  
object-creation-expression  
delegate-creation-expression  
anonymous-object-creation-expression  
typeof-expression  
 checked-expression  
unchecked-expression   
default-value-expression  
anonymous-method-expression

simple-name:  
identifier type-argument-listopt

parenthesized-expression:  
( expression )

member-access:  
primary-expression . identifier type-argument-listopt  
predefined-type . identifier type-argument-listopt  
qualified-alias-member . identifier

predefined-type: one of  
bool byte char decimal double float int long  
object sbyte short string uint ulong ushort

invocation-expression:  
primary-expression ( argument-listopt )

element-access:  
primary-no-array-creation-expression [ expression-list ]

expression-list:  
expression  
expression-list , expression

this-access:  
this

base-access:  
base . identifier  
base [ expression-list ]

post-increment-expression:  
primary-expression ++

post-decrement-expression:  
primary-expression --

object-creation-expression:  
new type ( argument-listopt ) object-or-collection-initializeropt   
new type object-or-collection-initializer

object-or-collection-initializer:  
object-initializer  
collection-initializer

object-initializer:  
{ member-initializer-listopt }  
{ member-initializer-list , }

member-initializer-list:  
member-initializer  
member-initializer-list , member-initializer

member-initializer:  
identifier = initializer-value

initializer-value:  
expression  
object-or-collection-initializer

collection-initializer:  
{ element-initializer-list }  
{ element-initializer-list , }

element-initializer-list:  
element-initializer  
element-initializer-list , element-initializer

element-initializer:  
non-assignment-expression  
{ expression-list }

array-creation-expression:  
new non-array-type [ expression-list ] rank-specifiersopt array-initializeropt  
new array-type array-initializer   
new rank-specifier array-initializer

delegate-creation-expression:  
new delegate-type ( expression )

anonymous-object-creation-expression:  
new anonymous-object-initializer

anonymous-object-initializer:  
{ member-declarator-listopt }  
{ member-declarator-list , }

member-declarator-list:  
member-declarator  
member-declarator-list , member-declarator

member-declarator:  
simple-name  
member-access  
identifier = expression

typeof-expression:  
typeof ( type )  
typeof ( unbound-type-name )  
typeof ( void )

unbound-type-name:  
identifier generic-dimension-specifieropt  
identifier :: identifier generic-dimension-specifieropt  
unbound-type-name . identifier generic-dimension-specifieropt

generic-dimension-specifier:  
< commasopt >

commas:  
,  
commas ,

checked-expression:  
checked ( expression )

unchecked-expression:  
unchecked ( expression )

default-value-expression:  
default ( type )

unary-expression:  
primary-expression  
+ unary-expression  
- unary-expression  
! unary-expression  
~ unary-expression  
pre-increment-expression  
pre-decrement-expression  
cast-expression

pre-increment-expression:  
++ unary-expression

pre-decrement-expression:  
-- unary-expression

cast-expression:  
( type ) unary-expression

multiplicative-expression:  
unary-expression  
multiplicative-expression \* unary-expression  
multiplicative-expression / unary-expression  
multiplicative-expression % unary-expression

additive-expression:  
multiplicative-expression  
additive-expression + multiplicative-expression  
additive-expression – multiplicative-expression

shift-expression:  
additive-expression   
shift-expression << additive-expression  
shift-expression right-shift additive-expression

relational-expression:  
shift-expression  
relational-expression < shift-expression  
relational-expression > shift-expression  
relational-expression <= shift-expression  
relational-expression >= shift-expression  
relational-expression is type  
relational-expression as type

equality-expression:  
relational-expression  
equality-expression == relational-expression  
equality-expression != relational-expression

and-expression:  
equality-expression  
and-expression & equality-expression

exclusive-or-expression:  
and-expression  
exclusive-or-expression ^ and-expression

inclusive-or-expression:  
exclusive-or-expression  
inclusive-or-expression | exclusive-or-expression

conditional-and-expression:  
inclusive-or-expression  
conditional-and-expression && inclusive-or-expression

conditional-or-expression:  
conditional-and-expression  
conditional-or-expression || conditional-and-expression

null-coalescing-expression:  
conditional-or-expression  
conditional-or-expression ?? null-coalescing-expression

conditional-expression:  
null-coalescing-expression  
null-coalescing-expression ? expression : expression

lambda-expression:  
anonymous-function-signature => anonymous-function-body

anonymous-method-expression:  
delegate explicit-anonymous-function-signatureopt block

anonymous-function-signature:  
explicit-anonymous-function-signature   
implicit-anonymous-function-signature

explicit-anonymous-function-signature:  
( explicit-anonymous-function-parameter-listopt )

explicit-anonymous-function-parameter-list:  
explicit-anonymous-function-parameter  
explicit-anonymous-function-parameter-list , explicit-anonymous-function-parameter

explicit-anonymous-function-parameter:  
anonymous-function-parameter-modifieropt type identifier

anonymous-function-parameter-modifier:   
ref  
out

implicit-anonymous-function-signature:  
( implicit-anonymous-function-parameter-listopt )  
implicit-anonymous-function-parameter

implicit-anonymous-function-parameter-list:  
implicit-anonymous-function-parameter  
implicit-anonymous-function-parameter-list , implicit-anonymous-function-parameter

implicit-anonymous-function-parameter:  
identifier

anonymous-function-body:  
expression  
block

query-expression:  
from-clause query-body

from-clause:  
from typeopt identifier in expression

query-body:  
query-body-clausesopt select-or-group-clause query-continuationopt

query-body-clauses:  
query-body-clause  
query-body-clauses query-body-clause

query-body-clause:  
from-clause  
let-clause  
where-clause  
join-clause  
join-into-clause  
orderby-clause

let-clause:  
let identifier = expression

where-clause:  
where boolean-expression

join-clause:  
join typeopt identifier in expression on expression equals expression

join-into-clause:  
join typeopt identifier in expression on expression equals expression into identifier

orderby-clause:  
orderby orderings

orderings:  
ordering  
orderings , ordering

ordering:  
expression ordering-directionopt

ordering-direction:  
ascending  
descending

select-or-group-clause:  
select-clause  
group-clause

select-clause:  
select expression

group-clause:  
group expression by expression

query-continuation:  
into identifier query-body

assignment:  
unary-expression assignment-operator expression

assignment-operator:  
=  
+=  
-=  
\*=  
/=  
%=  
&=  
|=  
^=  
<<=  
right-shift-assignment

expression:   
non-assignment-expression  
assignment

non-assignment-expression:  
conditional-expression  
lambda-expression  
query-expression  
backquote-expression

constant-expression:  
expression

boolean-expression:  
expression

backquote-expression:  
backquote-token  
expression backquote-token  
backquote-token expression

* + 1. Statements

statement:  
labeled-statement  
declaration-statement  
embedded-statement

embedded-statement:  
block  
empty-statement  
expression-statement  
selection-statement  
iteration-statement  
jump-statement  
try-statement  
checked-statement  
unchecked-statement  
lock-statement  
using-statement   
yield-statement

block:  
{ statement-listopt }

statement-list:  
statement  
statement-list statement

empty-statement:  
;

labeled-statement:  
identifier : statement

declaration-statement:  
local-variable-declaration ;  
local-constant-declaration ;

local-variable-declaration:  
local-variable-type local-variable-declarators

local-variable-type:  
type  
var

local-variable-declarators:  
local-variable-declarator  
local-variable-declarators , local-variable-declarator

local-variable-declarator:  
identifier  
identifier = local-variable-initializer

local-variable-initializer:  
expression  
array-initializer

local-constant-declaration:  
const type constant-declarators

constant-declarators:  
constant-declarator  
constant-declarators , constant-declarator

constant-declarator:  
identifier = constant-expression

expression-statement:  
statement-expression ;

statement-expression:  
invocation-expression  
object-creation-expression  
assignment  
post-increment-expression  
post-decrement-expression  
pre-increment-expression  
pre-decrement-expression

selection-statement:  
if-statement  
switch-statement

if-statement:  
if ( boolean-expression ) embedded-statement  
if ( boolean-expression ) embedded-statement else embedded-statement

switch-statement:  
switch ( expression ) switch-block

switch-block:  
{ switch-sectionsopt }

switch-sections:  
switch-section  
switch-sections switch-section

switch-section:  
switch-labels statement-list

switch-labels:  
switch-label  
switch-labels switch-label

switch-label:  
case constant-expression :  
default :

iteration-statement:  
while-statement  
do-statement  
for-statement  
foreach-statement

while-statement:  
while ( boolean-expression ) embedded-statement

do-statement:  
do embedded-statement while ( boolean-expression ) ;

for-statement:  
for ( for-initializeropt ; for-conditionopt ; for-iteratoropt ) embedded-statement

for-initializer:  
local-variable-declaration  
statement-expression-list

for-condition:  
boolean-expression

for-iterator:  
statement-expression-list

statement-expression-list:  
statement-expression  
statement-expression-list , statement-expression

foreach-statement:  
foreach ( local-variable-type identifier in expression ) embedded-statement

jump-statement:  
break-statement  
continue-statement  
goto-statement  
return-statement  
throw-statement

break-statement:  
break ;

continue-statement:  
continue ;

goto-statement:  
goto identifier ;  
goto case constant-expression ;  
goto default ;

return-statement:  
return expressionopt ;

throw-statement:  
throw expressionopt ;

try-statement:  
try block catch-clauses  
try block finally-clause  
try block catch-clauses finally-clause

catch-clauses:  
specific-catch-clauses general-catch-clauseopt  
specific-catch-clausesopt general-catch-clause

specific-catch-clauses:  
specific-catch-clause  
specific-catch-clauses specific-catch-clause

specific-catch-clause:  
catch ( class-type identifieropt ) block

general-catch-clause:  
catch block

finally-clause:  
finally block

checked-statement:  
checked block

unchecked-statement:  
unchecked block

lock-statement:  
lock ( expression ) embedded-statement

using-statement:  
using ( resource-acquisition ) embedded-statement

resource-acquisition:  
local-variable-declaration  
expression

yield-statement:  
yield return expression ;  
yield break ;

* + 1. Namespaces

compilation-unit:  
extern-alias-directivesopt using-directivesopt global-attributesopt namespace-member-declarationsopt

namespace-declaration:  
namespace qualified-identifier namespace-body ;opt

qualified-identifier:  
identifier  
qualified-identifier . identifier

namespace-body:  
{ extern-alias-directivesopt using-directivesopt namespace-member-declarationsopt }

extern-alias-directives:  
extern-alias-directive  
extern-alias-directives extern-alias-directive

extern-alias-directive:  
extern alias identifier ;

using-directives:  
using-directive  
using-directives using-directive

using-directive:  
using-alias-directive  
using-namespace-directive

using-alias-directive:  
using identifier = namespace-or-type-name ;

using-namespace-directive:  
using namespace-name ;

namespace-member-declarations:  
namespace-member-declaration  
namespace-member-declarations namespace-member-declaration

namespace-member-declaration:  
namespace-declaration  
type-declaration

type-declaration:  
class-declaration  
struct-declaration  
interface-declaration  
enum-declaration  
delegate-declaration

qualified-alias-member:  
identifier :: identifier type-argument-listopt

* + 1. Classes

class-declaration:  
attributesopt class-modifiersopt partialopt class identifier type-parameter-listopt class-baseopt type-parameter-constraints-clausesopt class-body ;opt

class-modifiers:  
class-modifier  
class-modifiers class-modifier

class-modifier:  
new  
public  
protected  
internal  
private  
abstract  
sealed  
static

type-parameter-list:  
< type-parameters >

type-parameters:  
attributesopt type-parameter  
type-parameters , attributesopt type-parameter

type-parameter:  
identifier

class-base:  
: class-type  
: interface-type-list  
: class-type , interface-type-list

interface-type-list:  
interface-type  
interface-type-list , interface-type

type-parameter-constraints-clauses:  
type-parameter-constraints-clause  
type-parameter-constraints-clauses type-parameter-constraints-clause

type-parameter-constraints-clause:  
where type-parameter : type-parameter-constraints

type-parameter-constraints:  
primary-constraint  
secondary-constraints  
constructor-constraint  
primary-constraint , secondary-constraints  
primary-constraint , constructor-constraint  
secondary-constraints , constructor-constraint  
primary-constraint , secondary-constraints , constructor-constraint

primary-constraint:  
class-type  
class  
struct

secondary-constraints:  
interface-type  
type-parameter  
secondary-constraints , interface-type  
secondary-constraints , type-parameter

constructor-constraint:  
new ( )

class-body:  
{ class-member-declarationsopt }

class-member-declarations:  
class-member-declaration  
class-member-declarations class-member-declaration

class-member-declaration:  
constant-declaration  
field-declaration  
method-declaration  
property-declaration  
event-declaration  
indexer-declaration  
operator-declaration  
constructor-declaration  
destructor-declaration  
static-constructor-declaration  
type-declaration

constant-declaration:  
attributesopt constant-modifiersopt const type constant-declarators ;

constant-modifiers:  
constant-modifier  
constant-modifiers constant-modifier

constant-modifier:  
new  
public  
protected  
internal  
private

constant-declarators:  
constant-declarator  
constant-declarators , constant-declarator

constant-declarator:  
identifier = constant-expression

field-declaration:  
attributesopt field-modifiersopt type variable-declarators ;

field-modifiers:  
field-modifier  
field-modifiers field-modifier

field-modifier:  
new  
public  
protected  
internal  
private  
static  
readonly  
volatile

variable-declarators:  
variable-declarator  
variable-declarators , variable-declarator

variable-declarator:  
identifier  
identifier = variable-initializer

variable-initializer:  
expression  
array-initializer

method-declaration:  
method-header method-body

method-header:  
attributesopt method-modifiersopt partialopt return-type member-name type-parameter-listopt ( formal-parameter-listopt ) type-parameter-constraints-clausesopt

method-modifiers:  
method-modifier  
method-modifiers method-modifier

method-modifier:  
new  
public  
protected  
internal  
private  
static  
virtual  
sealed  
override  
abstract  
extern

return-type:  
type  
void

member-name:  
identifier  
interface-type . identifier

method-body:  
block  
;

formal-parameter-list:  
fixed-parameters  
fixed-parameters , parameter-array  
parameter-array

fixed-parameters:  
fixed-parameter  
fixed-parameters , fixed-parameter

fixed-parameter:  
attributesopt parameter-modifieropt type identifier

parameter-modifier:  
ref  
out  
this

parameter-array:  
attributesopt params array-type identifier

property-declaration:  
attributesopt property-modifiersopt type member-name { accessor-declarations }

property-modifiers:  
property-modifier  
property-modifiers property-modifier

property-modifier:  
new  
public  
protected  
internal  
private  
static  
virtual  
sealed  
override  
abstract  
extern

member-name:  
identifier  
interface-type . identifier

accessor-declarations:  
get-accessor-declaration set-accessor-declarationopt  
set-accessor-declaration get-accessor-declarationopt

get-accessor-declaration:  
attributesopt accessor-modifieropt  get accessor-body

set-accessor-declaration:  
attributesopt accessor-modifieropt set accessor-body

accessor-modifier:  
protected  
internal  
private  
protected internal  
internal protected

accessor-body:  
block  
;

event-declaration:  
attributesopt event-modifiersopt event type variable-declarators ;  
attributesopt event-modifiersopt event type member-name { event-accessor-declarations }

event-modifiers:  
event-modifier  
event-modifiers event-modifier

event-modifier:  
new  
public  
protected  
internal  
private  
static  
virtual  
sealed  
override  
abstract  
extern

event-accessor-declarations:  
add-accessor-declaration remove-accessor-declaration  
remove-accessor-declaration add-accessor-declaration

add-accessor-declaration:  
attributesopt add block

remove-accessor-declaration:  
attributesopt remove block

indexer-declaration:  
attributesopt indexer-modifiersopt indexer-declarator { accessor-declarations }

indexer-modifiers:  
indexer-modifier  
indexer-modifiers indexer-modifier

indexer-modifier:  
new  
public  
protected  
internal  
private   
virtual  
sealed  
override  
abstract  
extern

indexer-declarator:  
type this [ formal-parameter-list ]  
type interface-type . this [ formal-parameter-list ]

operator-declaration:  
attributesopt operator-modifiers operator-declarator operator-body

operator-modifiers:  
operator-modifier  
operator-modifiers operator-modifier

operator-modifier:  
public  
static  
extern

operator-declarator:  
unary-operator-declarator  
binary-operator-declarator  
conversion-operator-declarator

unary-operator-declarator:  
type operator overloadable-unary-operator ( type identifier )

overloadable-unary-operator: one of  
+ - ! ~ ++ -- true false

binary-operator-declarator:  
type operator overloadable-binary-operator ( type identifier , type identifier )

overloadable-binary-operator:  
+  
-  
\*  
/  
%  
&  
|  
^  
<<  
right-shift  
==  
!=  
>  
<  
>=  
<=

conversion-operator-declarator:  
implicit operator type ( type identifier )  
explicit operator type ( type identifier )

operator-body:  
block  
;

constructor-declaration:  
attributesopt constructor-modifiersopt constructor-declarator constructor-body

constructor-modifiers:  
constructor-modifier  
constructor-modifiers constructor-modifier

constructor-modifier:  
public  
protected  
internal  
private  
extern

constructor-declarator:  
identifier ( formal-parameter-listopt ) constructor-initializeropt

constructor-initializer:  
: base ( argument-listopt )  
: this ( argument-listopt )

constructor-body:  
block  
;

static-constructor-declaration:  
attributesopt static-constructor-modifiers identifier ( ) static-constructor-body

static-constructor-modifiers:  
externopt static  
static externopt

static-constructor-body:  
block  
;

destructor-declaration:  
attributesopt externopt ~ identifier ( ) destructor-body

destructor-body:  
block  
;

* + 1. Structs

struct-declaration:  
attributesopt struct-modifiersopt partialopt struct identifier type-parameter-listopt struct-interfacesopt type-parameter-constraints-clausesopt struct-body ;opt

struct-modifiers:  
struct-modifier  
struct-modifiers struct-modifier

struct-modifier:  
new  
public  
protected  
internal  
private

struct-interfaces:  
: interface-type-list

struct-body:  
{ struct-member-declarationsopt }

struct-member-declarations:  
struct-member-declaration  
struct-member-declarations struct-member-declaration

struct-member-declaration:  
constant-declaration  
field-declaration  
method-declaration  
property-declaration  
event-declaration  
indexer-declaration  
operator-declaration  
constructor-declaration  
static-constructor-declaration  
type-declaration

* + 1. Arrays

array-type:  
non-array-type rank-specifiers

non-array-type:  
type

rank-specifiers:  
rank-specifier  
rank-specifiers rank-specifier

rank-specifier:  
[ dim-separatorsopt ]

dim-separators:  
,  
dim-separators ,

array-initializer:  
{ variable-initializer-listopt }  
{ variable-initializer-list , }

variable-initializer-list:  
variable-initializer  
variable-initializer-list , variable-initializer

variable-initializer:  
expression  
array-initializer

* + 1. Interfaces

interface-declaration:  
attributesopt interface-modifiersopt partialopt interface identifier type-parameter-listopt interface-baseopt type-parameter-constraints-clausesopt interface-body ;opt

interface-modifiers:  
interface-modifier  
interface-modifiers interface-modifier

interface-modifier:  
new  
public  
protected  
internal  
private

interface-base:  
: interface-type-list

interface-body:  
{ interface-member-declarationsopt }

interface-member-declarations:  
interface-member-declaration  
interface-member-declarations interface-member-declaration

interface-member-declaration:  
interface-method-declaration  
interface-property-declaration  
interface-event-declaration  
interface-indexer-declaration

interface-method-declaration:  
attributesopt newopt return-type identifier type-parameter-list ( formal-parameter-listopt ) type-parameter-constraints-clausesopt ;

interface-property-declaration:  
attributesopt newopt type identifier { interface-accessors }

interface-accessors:  
attributesopt get ;  
attributesopt set ;  
attributesopt get ; attributesopt set ;  
attributesopt set ; attributesopt get ;

interface-event-declaration:  
attributesopt newopt event type identifier ;

interface-indexer-declaration:  
attributesopt newopt type this [ formal-parameter-list ] { interface-accessors }

* + 1. Enums

enum-declaration:  
attributesopt enum-modifiersopt enum identifier enum-baseopt enum-body ;opt

enum-base:  
: integral-type

enum-body:  
{ enum-member-declarationsopt }  
{ enum-member-declarations , }

enum-modifiers:  
enum-modifier  
enum-modifiers enum-modifier

enum-modifier:  
new  
public  
protected  
internal  
private

enum-member-declarations:  
enum-member-declaration  
enum-member-declarations , enum-member-declaration

enum-member-declaration:  
attributesopt identifier  
attributesopt identifier = constant-expression

* + 1. Delegates

delegate-declaration:  
attributesopt delegate-modifiersopt delegate return-type identifier type-parameter-listopt ( formal-parameter-listopt ) type-parameter-constraints-clausesopt ;

delegate-modifiers:  
delegate-modifier  
delegate-modifiers delegate-modifier

delegate-modifier:  
new  
public  
protected  
internal  
private

* + 1. Attributes

global-attributes:  
global-attribute-sections

global-attribute-sections:  
global-attribute-section  
global-attribute-sections global-attribute-section

global-attribute-section:  
[ global-attribute-target-specifier attribute-list ]  
[ global-attribute-target-specifier attribute-list , ]

global-attribute-target-specifier:  
global-attribute-target :

global-attribute-target:  
assembly  
module

attributes:  
attribute-sections

attribute-sections:  
attribute-section  
attribute-sections attribute-section

attribute-section:  
[ attribute-target-specifieropt attribute-list ]  
[ attribute-target-specifieropt attribute-list , ]

attribute-target-specifier:  
attribute-target :

attribute-target:  
field  
event  
method  
param  
property  
return  
type

attribute-list:  
attribute  
attribute-list , attribute

attribute:  
attribute-name attribute-argumentsopt

attribute-name:  
 type-name

attribute-arguments:  
( positional-argument-listopt )  
( positional-argument-list , named-argument-list )  
( named-argument-list )

positional-argument-list:  
positional-argument  
positional-argument-list , positional-argument

positional-argument:  
attribute-argument-expression

named-argument-list:  
named-argument  
named-argument-list , named-argument

named-argument:  
identifier = attribute-argument-expression

attribute-argument-expression:  
expression