Elm

Elm
Application
Package

Application
ElmJson (ElmFile)

Package
ElmJson (ElmFile)

ElmFile
Module

Module
Definition (Import) (Declaration)

Definition
(port) module ModuleName exposing ((.. | ExposedItem (, ExposedItem)))

Import
import ModuleName (as ModuleAliasName) (exposing (.. | ExposedItem (, ExposedItem)))

ExposedItem
TypeName
TypeName(..)
ValueName
(OperatorSymbol)

Declaration
type TypeName (TypeVariableName) = TypeVariantName (SingleTypeExpression) (| TypeVariantName (SingleTypeExpression))

type alias TypeName (TypeVariableName) = TypeExpression

ValueName : TypeExpression

ValueName (DestructingPattern) = Expression

port ValueName : TypeExpression

Type

TypeExpression

TypeName (ModuleName.Typename (SingleTypeExpression)) (-> TypeExpression)
SingleTypeExpression

SingleTypeExpression

(ModuleName.) TypeName
TypeVariableName
ConstrainedType
()
(TypeExpression, TypeExpression)
(TypeExpression, TypeExpression, TypeExpression)
{}
{ ({ TypeVariableName |) FieldName : TypeExpression (, FieldName : TypeExpression) }
(TypeExpression)

Name

ModuleName
(UppercaseName (. UppercaseName))

ModuleAliasName
UppercaseName

TypeName
UppercaseName

TypeVariantName
UppercaseName

TypeVariableName
LowercaseName

ConstrainedType
number
appendable
comparable
compappend

ValueName
LowercaseName

FieldName
LowercaseName

UppercaseName
[A-Z][0-9A-Za-z_]*

LowercaseName
[a-z][0-9A-Za-z_]*

Value

Expression

let (ValueName (DestructingPattern) = Expression | DestructingPattern = Expression | ValueName : TypeExpression) in Expression

(if Expression then Expression else Expression | else Expression)

case Expression of (Pattern -> Expression)

\ (DestructingPattern) -> Expression

(SingleExpression | OperatorSymbol)

SingleExpression

Number
'Char'
"String"
[]
[(Expression (, Expression))]
()
(Expression, Expression)
(Expression, Expression, Expression)
{}
{ ({ ValueName |) FieldName = Expression (, FieldName = Expression) } FieldAccessor
[glsl| OpenGlShadingLanguage |]
(ModuleName.) TypeVariantName
(ModuleName.) ValueName FieldAccessor
.FieldName
(OperatorSymbol)
-(Expression)
(Expression) FieldAccessor

FieldAccessor

(.FieldName)

Pattern

(SinglePattern (as ValueName) (:: SingleTypePattern (as ValueName)))

SinglePattern

ValueName
_
Number
'Char'
"String"
[]
[(Pattern (, Pattern))]
()
(Pattern, Pattern)
(Pattern, Pattern, Pattern)
{}
{ (FieldName (, FieldName)) }
(ModuleName.) TypeVariantName (SinglePattern)
(Pattern)

DestructingPattern

ValueName
_
()
(DestructingPattern, DestructingPattern)
(DestructingPattern, DestructingPattern, DestructingPattern)
{}
{ (FieldName (, FieldName)) }
(ModuleName.) TypeVariantName
((ModuleName.) TypeVariantName (DestructingPattern))
(DestructingPattern as ValueName)
(DestructingPattern)