# Changes to 13 (“ECMAScript Language: Statements and Declarations”)

**Syntax**

*Declaration*[Default, Yield] :

*FunctionDeclaration*[?Default]

*GeneratorDeclaration*[?Default]

*ClassDeclaration*[?Default]

*LexicalDeclaration*[In, ?Yield]

# Changes to 14 (“ECMAScript Language: Functions and Classes”)

## Changes to 14.1 (“Function Definitions”)

**Syntax**

*FunctionDeclaration*[Default] :

**function** *BindingIdentifier*[?Default] **(** *FormalParameters* **)** **{** *FunctionBody* **}**

## Changes to 14.4 (“Generator Function Definitions”)

**Syntax**

*GeneratorDeclaration*[Default] :

**function** **\*** *BindingIdentifier*[?Default] **(** *FormalParameters* **)** **{** *FunctionBody* **}**

## Changes to 14.5 (“Class Definitions”)

**Syntax**

*ClassDeclaration*[Default] :

**class** *BindingIdentifier*[?Default] *ClassTail*

# Changes to 15.1 (“Modules”)

**Syntax**

*Module* :

*ModuleBodyopt*

*ModuleBody* :

*ModuleItemList*

*ModuleItemList* :

*ModuleItem*

*ModuleItemList* *ModuleItem*

*ModuleItem* :

*ModuleDeclaration*

*ImportDeclaration*

*ExportDeclaration*

*StatementListItem*

*ModuleDeclaration* :

**module** [no *LineTerminator* here] *ImportedBinding* *FromClause* **;**

*ImportedBinding* :

*BindingIdentifier*

*FromClause* :

**from** *ModuleSpecifier*

*ModuleSpecifier* :

*StringLiteral*

*ImportDeclaration* :

**import** *ImportClause* *FromClause* ;

**import** *ModuleSpecifier* **;**

*ImportClause* :

*ImportedBinding*

*ImportedBinding* **,** *NamedImports*

*NamedImports*

*NamedImports* :

**{** **}**

**{** *ImportsList* **}**

**{** *ImportsList* **,** **}**

*ImportsList* :

*ImportSpecifier*

*ImportsList* **,** *ImportSpecifier*

*ImportSpecifier* :

*ImportedBinding*

*IdentifierName* **as** *ImportedBinding*

*ExportDeclaration* :

**export** *ExportsClause* *FromClauseopt* **;**

**export** **\*** *FromClause* **;**

**export** *VariableStatement*

**export** *Declaration*[Default]

**export** **default** *AssignmentExpression* **;**

*ExportsClause* :

**{** **}**

**{** *ExportsList* **}**

**{** *ExportsList* **,** **}**

*ExportsList* :

*ExportSpecifier*

*ExportsList* **,** *ExportSpecifier*

*ExportSpecifier* :

*IdentifierReference*

*IdentifierReference* **as** *IdentifierName*

**Static Semantics: Early Errors**

*ModuleBody* : *ModuleItemList*

* It is a Syntax Error if the LexicallyDeclaredNames of *ModuleItemList* contains any duplicate entries.
* It is a Syntax Error if any element of the LexicallyDeclaredNames of *ModuleItemList* also occurs in the VarDeclaredNames of *ModuleItemList*.
* It is a Syntax Error if *ModuleItemList* contains *ReturnStatement*.
* It is a Syntax Error if *ModuleItemList* contains **super**.

*ImportedBinding* : *BindingIdentifier*

* It is a Syntax Error if the BoundNames of *BindingIdentifier* contains the string **"let"**, **"eval"**, or **"arguments"**.

**~~Static Semantics: ImportedNames~~**

**Static Semantics: IsStrict**

*ModuleBody* : *ModuleItemList*

1. Return **true**.

**Static Semantics: LexicallyDeclaredNames**

*ModuleItemList* : *ModuleItemList* *ModuleItem*

1. Let *names* be the LexicallyDeclaredNames of *ModuleItemList*.
2. Append to *names* the LexicallyDeclaredNames of *ModuleItem*.
3. Return *names*.

*ModuleItem* : *ModuleDeclaration*

1. Return the BoundNames of *ModuleDeclaration*.

*ModuleItem* : *ImportDeclaration*

1. Return the BoundNames of *ImportDeclaration*.

*ModuleItem* : *ExportDeclaration*

1. Return the LexicallyDeclaredNames of *ExportDeclaration*.

*ExportDeclaration* : **export** *ExportsList* *FromClauseopt* **;**

*ExportDeclaration* : **export** **\*** *FromClause* **;**

*ExportDeclaration* : **export** *VariableStatement*

1. Return a new empty List.

*ExportDeclaration* : **export** *Declaration*

1. Return the BoundNames of *Declaration*.

*ExportDeclaration* : **export** **default** *AssignmentExpression* **;**

1. Return a new List containing **"default"**.

*ModuleItem* : *StatementListItem*

1. Return the TopLevelLexicallyDeclaredNames of *StatementListItem*.

**Static Semantics: LexicallyScopedDeclarations**

*ModuleItemList* : *ModuleItemList* *ModuleItem*

1. Let *declarations* be the LexicallyScopedDeclarations of *ModuleItemList*.
2. Append to *declarations* the LexicallyScopedDeclarations of *ModuleItem*.
3. Return *declarations*.

*ModuleItem* : *ModuleDeclaration*

1. Return a new List containing *ModuleDeclaration*.

*ModuleItem* : *ImportDeclaration*

1. Return a new List containing *ImportDeclaration*.

*ModuleItem* : *ExportDeclaration*

1. Return the LexicallyScopedDeclarations of *ExportDeclaration*.

*ExportDeclaration* : **export** *ExportsClause* *FromClauseopt* **;**

*ExportDeclaration* : **export** **\*** *FromClause* **;**

*ExportDeclaration* : **export** *VariableStatement*

1. Return a new empty List.

*ExportDeclaration* : **export** *Declaration*

1. Return a new List containing *Declaration*.

*ExportDeclaration* : **export** **default** *AssignmentExpression* **;**

1. Return a new List containing *ExportDeclaration*.

*ModuleItem* : *StatementListItem*

1. Return the TopLevelLexicallyScopedDeclarations of *StatementListItem*.

**Static Semantics: VarDeclaredNames**

*ModuleItemList* : *ModuleItemList* *ModuleItem*

1. Let *names* be the VarDeclaredNames of *ModuleItemList*.
2. Append to *names* the VarDeclaredNames of *ModuleItem*.
3. Return *names*.

*ModuleItem* : *ModuleDeclaration*

*ModuleItem* : *ImportDeclaration*

1. Return a new empty List.

*ModuleItem* : *ExportDeclaration*

1. Return the VarDeclaredNames of *ExportDeclaration*.

*ExportDeclaration* : **export** *VariableStatement*

1. Return the VarDeclaredNames of *VariableStatement*.

*ExportDeclaration* : **export** *ExportsClause* *FromClauseopt* **;**

*ExportDeclaration* : **export** **\*** *FromClause* **;**

*ExportDeclaration* : **export** *Declaration*

*ExportDeclaration* : **export** **default** *AssignmentExpression* **;**

1. Return a new empty List.

*ModuleItem* : *StatementListItem*

1. Return the TopLevelVarDeclaredNames of *StatementListItem*.

**Static Semantics: VarScopedDeclarations**

*ModuleItemList* : *ModuleItemList* *ModuleItem*

1. Let *declarations* be the VarScopedDeclarations of *ModuleItemList*.
2. Append to *declarations* the VarScopedDeclarations of *ModuleItem*.
3. Return *declarations*.

*ModuleItem* : *ModuleDeclaration*

*ModuleItem* : *ImportDeclaration*

1. Return a new empty List.

*ModuleItem* : *ExportDeclaration*

1. Return the VarScopedDeclarations of *ExportDeclaration*.

*ExportDeclaration* : **export** *ExportsClause* *FromClauseopt***;**

*ExportDeclaration* : **export** **\*** *FromClause* **;**

*ExportDeclaration* : **export** *Declaration*

*ExportDeclaration* : **export** **default** *AssignmentExpression* **;**

1. Return a new empty List.

*ExportDeclaration* : **export** *VariableStatement*

1. Return a new List containing *VariableStatement*.

*ModuleItem* : *StatementListItem*

1. Return the TopLevelLexicallyScopedDeclarations of *StatementListItem*.

**Static Semantics: BoundNames**

*ModuleDeclaration* : **module** [no *LineTerminator* here] *ImportedBinding* *FromClause* **;**

1. Let *names* be a new List containing *ImportedBinding*.
2. Return *names*.

*ImportDeclaration* : **import** *ImportsClause* *FromClause* **;**

1. Return the BoundNames of *ImportsClause*.

*ImportsClause* : *BindingIdentifier*

1. Let *names* be a new List containing *BindingIdentifier*.
2. Return *names*.

*ImportsClause* : *BindingIdentifier* **,** *NamedImports*

1. Let *names* be a new List containing *BindingIdentifier*.
2. Let *specs* be the ImportSpecs of *ImportSpecifierSet*.
3. For each *spec* in *specs*, do
   1. Append *spec*.[[LocalName]] to *names*.
4. Return *names*.

*ImportsClause* : *NamedImports*

1. Let *specs* be the ImportSpecs of *ImportSpecifierSet*.
2. Let *names* be a new empty List.
3. For each *spec* in *specs*, do
   1. Add *spec*.[[LocalName]] to *names*.
4. Return *names*.

*ImportDeclaration* : **import** *ModuleSpecifier* **;**

1. Return a new empty List.

*ExportDeclaration* : **export** *VariableStatement*

1. Let *names* be the BoundNames of *VariableStatement*.
2. Return *names*.

*ExportDeclaration* : **export** *Declaration*

1. Let *names* be the BoundNames of *Declaration*.
2. Return *names*.

*ExportDeclaration* : **export** **default** *AssignmentExpression* **;**

1. Let *names* be a new List containing the string **"default"**.
2. Return *names*.

**Static Semantics: ImportSpecs**

*NamedImports* : **{** **}**

1. Return a new empty List.

*NamedImports* : **{** *ImportsList* **}**

*NamedImports* : **{** *ImportsList* **,** **}**

1. Return the ImportSpecs of *ImportsList*.

*ImportsList* : *ImportSpecifier*

1. Return the ImportSpecs of *ImportSpecifier*.

*ImportsList* : *ImportsList* **,** *ImportSpecifier*

1. Let *specs* be the ImportSpecs of *ImportsList*.
2. Append the ImportSpecs of *ImportSpecifier* to *specs*.
3. Return *specs*.

*ImportSpecifier* : *ImportedBinding*

1. Return a new List containing the record {[[LocalName]]: *ImportedBinding*, [[ImportName]]: *ImportedBinding*}.

*ImportSpecifier* : *IdentifierName* **as** *ImportedBinding*

1. Return a new List containing the record {[[LocalName]]: *ImportedBinding*, [[ImportName]]: *IdentifierName*}.

**Static Semantics: ExportSpecs**

*ExportsClause* : **{** **}**

1. Return a new empty List.

*ExportsClause* : **{** *ExportsList* **}**

*ExportsClause* : **{** *ExportsList* **,** **}**

1. Return the ExportSpecs of *ExportsList*.

*ExportsList* : *ExportSpecifier*

1. Return the ExportSpecs of *ExportSpecifier*.

*ExportsList* : *ExportsList* **,** *ExportSpecifier*

1. Let *specs* be the ExportSpecs of *ExportsList*.
2. Append the ExportSpecs of *ExportSpecifier* to *specs*.
3. Return *specs*.

*ExportSpecifier* : *IdentifierReference*

1. Return a new List containing the record {[[LocalName]]: *IdentifierReference*, [[ExportName]]: *IdentifierReference*}.

*ExportSpecifier* : *IdentifierReference* **as** *IdentifierName*

1. Return a new List containing the record {[[LocalName]]: *IdentifierReference*, [[ExportName]]: *IdentifierName*}.

**Static Semantics: ModuleRequests**

*ModuleItemList* : [empty]

1. Return a new empty List.

*ModuleItemList* : *ModuleItemList* *ModuleItem*

1. Let *requests* be the ModuleRequests of *ModuleItemList*.
2. Append the ModuleRequests of *ModuleItem* to *requests*.
3. Return *requests*.

*ModuleDeclaration* : **module** [no *LineTerminator* here] *BindingIdentifier* *FromClause* **;**

*ImportDeclaration* : **import** *ImportsClause* *FromClause* **;**

*ExportDeclaration* : **export** *ExportsClause* *FromClause* **;**

*ExportDeclaration* : **export** **\*** *FromClause* **;**

1. Return the ModuleRequests of *FromClause*.

*ImportDeclaration* : **import** *ModuleSpecifier* **;**

1. Return a new List containing *ModuleSpecifier*.

*ExportDeclaration* : **export** *ExportsClause* **;**

*ModuleItem* : *StatementListItem*

1. Return a new empty List.

*FromClause* : **from** *ModuleSpecifier*

1. Return a new List containing *ModuleSpecifier*.

**Static Semantics: KnownExportEntries**

*ExportDeclaration* : **export** *ExportsClause* **;**

1. Let *entries* be a new empty List.
2. Let *specs* be the ExportSpecs of *ExportsClause*.
3. For each *spec* in *specs*, do
   1. Add the record {[[ModuleRequest]]: **null**, [[ImportName]]: **null**, [[LocalName]]: *spec*.[[LocalName]], [[ExportName]]: *spec*.[[ExportName]]} to *entries*.
4. Return *entries*.

*ExportDeclaration* : **export** *ExportsClause* **from** *ModuleSpecifier* **;**

1. Let *entries* be a new empty List.
2. Let *specs* be the ExportSpecs of *ExportsClause*.
3. For each *spec* in *specs*, do
   1. Add the record {[[ModuleRequest]]: *ModuleSpecifier*, [[ImportName]]: *spec*.[[LocalName]], [[LocalName]]: **null**, [[ExportName]]: *spec*.[[ExportName]]} to *entries*.
4. Return *entries*.

*ExportDeclaration* : **export** *VariableStatement*

1. Let *entries* be a new empty List.
2. Let *names* be the BoundNames of *VariableStatement*.
3. For each *name* in *names*, do
   1. Add the record {[[ModuleRequest]]: **null**, [[ImportName]]: **null**, [[LocalName]]: *name*, [[ExportName]]: *name*} to *entries*.
4. Return *entries*.

*ExportDeclaration* : **export** *Declaration*

1. Let *entries* be a new empty List.
2. Let *names* be the BoundNames of *Declaration*.
3. For each *name* in *names*, do
   1. Add the record {[[ModuleRequest]]: **null**, [[ImportName]]: **null**, [[LocalName]]: *name*, [[ExportName]]: *name*} to *entries*.
4. Return *entries*.

*ExportDeclaration* : **export** **default** *AssignmentExpression* **;**

1. Let *entries* be a new List containing the record {[[ModuleRequest]]: **null**, [[ImportName]]: **null**, [[LocalName]]: **"default"**, [[ExportName]]: **"default"**}.
2. Return *entries*.

**Static Semantics: UnknownExportEntries**

*ExportDeclaration* : **export** **\*****from** *ModuleSpecifier* **;**

1. Return a new List containing the record {[[ModuleRequest]]: *ModuleSpecifier*, [[ImportName]]: **all**, [[LocalName]]: **null**, [[ExportName]]: **all**}.

**Static Semantics: ImportEntries**

*ModuleDeclaration* : **module** [no *LineTerminator* here] *ImportedBinding* **from** *ModuleSpecifier* **;**

1. Return a new List containing the record {[[ModuleRequest]]: *ModuleSpecifier*, [[ImportName]]: **module**, [[LocalName]]: *ImportedBinding*}.

*ImportDeclaration* : **import** *ImportedBinding* **from** *ModuleSpecifier* **;**

1. Let *entries* be a new List containing the record {[[ModuleRequest]]: *ModuleSpecifier*, [[ImportName]]: **"default"**, [[LocalName]]: *ImportedBinding*}.
2. Return *entries*.

*ImportDeclaration* : **import** *ImportedBinding* **,** *NamedImports* **from** *ModuleSpecifier* **;**

1. Let *entries* be a new List containing the record {[[ModuleRequest]]: *ModuleSpecifier*, [[ImportName]]: **"default"**, [[LocalName]]: *ImportedBinding*}.
2. Let *specs* be the ImportSpecs of *NamedImports*.
3. For each *spec* of *specs*, do
   1. Add the record {[[ModuleRequest]]: *ModuleSpecifier*, [[ImportName]]: *spec*.[[ImportName]], [[LocalName]]: *spec*.[[LocalName]]} to *entries*.
4. Return *entries*.

*ImportDeclaration* : **import** *NamedImports* **from** *ModuleSpecifier* **;**

1. Let *entries* be a new empty List.
2. Let *specs* be the ImportSpecs of *NamedImports*.
3. For each *spec* of *specs*, do
   1. Add the record {[[ModuleRequest]]: *ModuleSpecifier*, [[ImportName]]: *spec*.[[ImportName]], [[LocalName]]: *spec*.[[LocalName]]} to *entries*.
4. Return *entries*.

# Changes to 15.2 (“Scripts”)

**Syntax**

*Script* :

*ScriptBodyopt*

*ScriptBody* :

*StatementList*