4.3 Identifiers

When an expression is an *Identifier*, the expression refers to the most nested module, class, enum, function, variable, or parameter with that name whose scope (section 2.4) includes the location of the reference. The type of such an expression is the type associated with the referenced entity:

- For a module, the object type associated with the module instance.
- For a class, the constructor type associated with the constructor function object.
- For an enum, the object type associated with the enum object.
- For a function, the function type associated with the function object.
- For a variable, the type of the variable.
- For a parameter, the type of the parameter.

An identifier expression that references a variable or parameter is classified as a reference. An identifier expression that references any other kind of entity is classified as a value (and therefore cannot be the target of an assignment).

4.4 Literals

Literals are typed as follows:

- The type of the `null` literal is the Null primitive type.
- The type of the literals `true` and `false` is the Boolean primitive type.
- The type of numeric literals is the Number primitive type.
- The type of string literals is the String primitive type.
- The type of regular expression literals is the global interface type 'RegExp'.

4.5 Object Literals

Object literals are extended to support type annotations in get and set accessors.

PropertyAssignment: (Modified)
PropertyName `:` *AssignmentExpression*
PropertyName *CallSignature* `{` *FunctionBody* `}`
GetAccessor
SetAccessor

GetAccessor:
`get` *PropertyName* `(` `)` $TypeAnnotation_{opt}$ `{` *FunctionBody* `}`

SetAccessor:
`set` *PropertyName* `(` *Identifier* $TypeAnnotation_{opt}$ `)` `{` *FunctionBody* `}`

The type of an object literal is an object type with the set of properties specified by the property assignments in the object literal. A get and set accessor may specify the same property name, but otherwise it is an error to specify multiple property assignments for the same property.