5 Statements

This chapter describes the static type checking TypeScript provides for JavaScript statements. TypeScript itself does not introduce any new statement constructs.

5.1 Variable Statements

Variable statements are extended to include optional type annotations.

VariableDeclaration: (Modified)
    Identifier *TypeAnnotation*$_{opt}$ *Initialiser*$_{opt}$

VariableDeclarationNoIn: (Modified)
    Identifier *TypeAnnotation*$_{opt}$ *InitialiserNoIn*$_{opt}$

TypeAnnotation:
    `:` *Type*

A variable declaration introduces a variable with the given name in the containing declaration space. The type associated with a variable is determined as follows:

- If the declaration includes a type annotation, the stated type becomes the type of the variable. If an initializer is present, the initializer expression is contextually typed (see section 4.19) by the stated type and must be assignable to the stated type, or otherwise a compile-time error occurs.
- If the declaration includes an initializer but no type annotation, the widened type (see section 3.9) of the initializer expression becomes the type of the property.
- If the declaration includes neither a type annotation nor an initializer, the type of the variable becomes the Any type.

TODO: Specify what type results when a variable initializer directly or indirectly references the variable it is initializing.

Multiple declarations for the same variable name in the same declaration space are permitted, provided that each declaration associates the same type with the variable.

Below are some examples of variable declarations and their associated types.

```
var a;                          // any
var b: number;                  // number
var c = 1;                      // number
var d = { x: 1, y: "hello" };   // { x: number; y: string; }
var e: any = "test";            // any
```

The following is permitted because all declarations of the single variable ‘x’ associate the same type (Number) with ‘x’.