The hierarchy formed by namespace and named type names partially mirrors that formed by module instances and members. The example

```
module A {
    export module B {
        export class C { }
    }
}
```

introduces a named type with the qualified name 'A.B.C' and also introduces a constructor function that can be accessed using the expression 'A.B.C'. Thus, in the example

```
var c: A.B.C = new A.B.C();
```

the two occurrences of 'A.B.C' in fact refer to different entities. It is the context of the occurrences that determines whether 'A.B.C' is processed as a type name or an expression.

2.3 Declarations

Declarations introduce names in the ***declaration spaces*** to which they belong. It is an error to have two names with same spelling in the same declaration space. Declaration spaces exist as follows:

- The global module and each external or internal module has a declaration space for variables (including functions, modules, class constructor functions, and enum objects), a declaration space for named types (classes, interfaces, and enums), and a declaration space for namespaces (containers of named types). Every declaration (whether local or exported) in a module contributes to one or more of these declaration spaces.
- Each external or internal module has a declaration space for exported members, a declaration space for exported named types, and a declaration space for exported namespaces. All export declarations in the module contribute to these declaration spaces. Each internal module's export declaration spaces are shared with other internal modules that have the same root module and the same qualified name starting from that root module.
- Each class declaration has a declaration space for instance members, a declaration space for static members, and a declaration space for type parameters.
- Each interface declaration has a declaration space for members and a declaration space for type parameters. An interface's declaration space is shared with other interfaces that have the same root module and the same qualified name starting from that root module.
- Each enum declaration has a declaration space for its enum members. An enum's declaration space is shared with other enums that have the same root module and the same qualified name starting from that root module.
- Each function declaration (including constructor, member function, and member accessor declarations) and each function expression has a declaration space for variables (parameters, local variables, and local functions) and a declaration space for type parameters.
- Each object literal has a declaration space for its properties.
- Each object type literal has a declaration space for its members.