'C:/src/ui/main', the external module names './editor' and '../lib/io' reference modules with the file paths 'C:/src/ui/editor' and 'C:/src/lib/io'.

- If the import declaration specifies a top-level external module name and the program contains an *AmbientExternalModuleDeclaration* (section 12.1.6) with a string literal that specifies that exact name, then the import declaration references that ambient external module.
- If the import declaration specifies a top-level external module name and the program contains no *AmbientExternalModuleDeclaration* (section 12.1.6) with a string literal that specifies that exact name, the name is resolved in a host dependent manner (for example by considering the name relative to a module name space root). If a matching module cannot be found an error occurs.

11.2.2 External Import Declarations

External import declarations are used to import external modules and create local aliases by which they may be referenced.

ExternalImportDeclaration:
    `import` *Identifier* `=` *ExternalModuleReference* `;`

ExternalModuleReference:
    `require` `(` *StringLiteral* `)`

The string literal specified in an *ExternalModuleReference* is interpreted as an external module name (section 11.2.1).

An external import declaration introduces a local identifier that references a given external module. The local identifier becomes an alias for, and is classified exactly like, the entity or entities exported from the referenced external module. Specifically, if the referenced external module contains no export assignment the identifier is classified as a module, and if the referenced external module contains an export assignment the identifier is classified exactly like the entity or entities named in the export assignment.

11.2.3 Export Declarations

An external module that contains no export assignment (section 11.2.4) exports an entity classified as a module. Similarly to an internal module, export declarations (section 10.4) in the external module are used to declare the members of this entity.

Unlike a non-instantiated internal module (section 10.1), an external module containing only interface types and non-instantiated internal modules still has a module instance associated with it, albeit one with no members.

If an external module contains an export assignment it is an error for the external module to also contain export declarations. The two types of exports are mutually exclusive.

11.2.4 Export Assignments

An export assignment designates a module member as the entity to be exported in place of the external module itself.