Furthermore, when a function expression is contextually typed by a function type *T*, expressions in contained return statements (section 5.7) are contextually typed by *T*'s return type.

4.10 Property Access

A property access uses either dot notation or bracket notation. A property access expression is always classified as a reference.

A property access uses an object's apparent type (section 3.8.1) to determine its properties. Furthermore, in a property access, an object's apparent type includes the properties that originate in the 'Object' or 'Function' global interface types, as described in section 3.3.

A dot notation property access of the form

> *ObjExpr* **.** *Name*

where *ObjExpr* is an expression and *Name* is an identifier (including, possibly, a reserved word), is used to access the property with the given name on the given object. A dot notation property access is processed as follows at compile-time:

- If *ObjExpr* is of type Any, any *Name* is permitted and the property access is of type Any.
- Otherwise, if *Name* denotes a property member in the apparent type of *ObjExpr*, the property access is of the type of that property.
- Otherwise, the property access is invalid and a compile-time error occurs.

A bracket notation property access of the form

> *ObjExpr* [*IndexExpr*]

where *ObjExpr* and *IndexExpr* are expressions, is used to access the property with the name computed by the index expression on the given object. A bracket notation property access is processed as follows at compile-time:

- If *IndexExpr* is a string literal or a numeric literal and *ObjExpr*'s apparent type has a property with the name given by that literal (converted to its string representation in the case of a numeric literal), the property access is of the type of that property.
- Otherwise, if *ObjExpr*'s apparent type has a numeric index signature and *IndexExpr* is of type Any, the Number primitive type, or an enum type, the property access is of the type of that index signature.
- Otherwise, if *ObjExpr*'s apparent type has a string index signature and *IndexExpr* is of type Any, the String or Number primitive type, or an enum type, the property access is of the type of that index signature.
- Otherwise, if *IndexExpr* is of type Any, the String or Number primitive type, or an enum type, the property access is of type Any.
- Otherwise, the property access is invalid and a compile-time error occurs.