or by writing the element type as an object type literal

```
{ (): string }[]
```

3.7 Object Type Literals

An object type literal defines an object type by specifying the set of members that are statically considered to be present in instances of the type. Object type literals can be given names using interface declarations but are otherwise anonymous.

ObjectType:
  `{` *TypeBody*$_{opt}$ `}`

TypeBody:
  TypeMemberList `;`$_{opt}$

TypeMemberList:
  TypeMember
  TypeMemberList `;` *TypeMember*

TypeMember:
  PropertySignature
  CallSignature
  ConstructSignature
  IndexSignature
  MethodSignature

The members of an object type literal are specified as a combination of property, call, construct, index, and method signatures. The signatures are separated by semicolons and enclosed in curly braces.

3.7.1 Property Signatures

A property signature declares the name and type of a property member.

PropertySignature:
  PropertyName `?`$_{opt}$ *TypeAnnotation*$_{opt}$

PropertyName:
  IdentifierName
  StringLiteral
  NumericLiteral

The *PropertyName* production, reproduced above from the ECMAScript grammar, permits a property name to be any identifier (including a reserved word), a string literal, or a numeric literal. String literals can be used to give properties names that are not valid identifiers, such as names containing blanks. Numeric literal property names are equivalent to string literal property names with the string representation of the numeric literal, as defined in the ECMAScript specification.