Dictionaries

On this page:

Dictionary Syntax and Semantics

A dictionary is a mapping from a key type to a value type.

For example:

Slice

```
struct Employee {
    long   number;
    string firstName;
    string lastName;
};

dictionary<long, Employee> EmployeeMap;
```

This definition creates a dictionary named `EmployeeMap` that maps from an employee number to a structure containing the details for an employee. Whether or not the key type (the employee number, of type `long` in this example) is also part of the value type (the `Employee` structure in this example) is up to you — as far as Slice is concerned, there is no need to include the key as part of the value.

Dictionaries can be used to implement sparse arrays, or any lookup data structure with non-integral key type. Even though a sequence of structures containing key-value pairs could be used to model the same thing, a dictionary is more appropriate:

- A dictionary clearly signals the intent of the designer, namely, to provide a mapping from a domain of values to a range of values. (A sequence of structures of key-value pairs does not signal that same intent as clearly.)
- At the programming language level, sequences are implemented as vectors (or possibly lists), that is, they are not well suited to model sparsely populated domains and require a linear search to locate an element with a particular value. On the other hand, dictionaries are implemented as a data structure (typically a hash table or red-black tree) that supports efficient searching in $O(\log n)$ average time or better.

Allowable Types for Dictionary Keys and Values

The key type of a dictionary need not be an integral type. For example, we could use the following definition to translate the names of the days of the week:

Slice

```
dictionary<string, string> WeekdaysEnglishToGerman;
```

The server implementation would take care of initializing this map with the key-value pairs `Monday-Montag`, `Tuesday-Dienstag`, and so on.

The value type of a dictionary can be any Slice type. However, the key type of a dictionary is limited to one of the following types:

- Integral types (`byte`, `short`, `int`, `long`, `bool`)
- `string`
- `enum`
- Structures containing only data members of integral type or `string`

Complex nested types, such as nested structures, sequences, or dictionaries, and floating-point types (`float` and `double`) cannot be used as the key type. Complex nested types are disallowed because they complicate the language mappings for dictionaries, and floating-point types are disallowed because representational changes of values as they cross machine boundaries can lead to ill-defined semantics for equality.

See Also