Slice Checksums

As distributed applications evolve, developers and system administrators must be careful to ensure that deployed components are using the same client-server contract. Unfortunately, mistakes do happen, and it is not always readily apparent when they do.

To minimize the chances of this situation, the Slice compilers support an option that generates checksums for Slice definitions, thereby enabling two peers to verify that they share an identical client-server contract. The checksum for a Slice definition includes details such as parameter and member names and the order in which operations are defined, but ignores information that is not relevant to the client-server contract, such as metadata, comments, and formatting.

This option causes the Slice compiler to construct a dictionary that maps Slice type identifiers to checksums. A server typically supplies an operation that returns its checksum dictionary for the client to compare with its local version, at which point the client can take action if it discovers a mismatch.

The dictionary type is defined in the file `Ice/SliceChecksumDict.ice` as follows:

Slice

```
module Ice {
    dictionary<string, string> SliceChecksumDict;
};
```

This type can be incorporated into an application's Slice definitions like this:

Slice

```
#include <Ice/SliceChecksumDict.ice>

interface MyServer {
    idempotent Ice::SliceChecksumDict getSliceChecksums();
    // ...
};
```

The key of each element in the dictionary is a Slice type ID, and the value is the checksum of that type.

For more information on generating and using Slice checksums, see the appropriate language mapping chapter.

See Also

- Type IDs