Using Slice Checksums in C++

The Slice compilers can optionally generate checksums of Slice definitions. For `slice2cpp`, the `--checksum` option causes the compiler to generate code in each C++ source file that accumulates checksums in a global map. A copy of this map can be obtained by calling a function defined in the header file `Ice/SliceChecksums.h`:

C++

```
namespace Ice {
    Ice::SliceChecksumDict sliceChecksums();
}
```

In order to verify a server's checksums, a client could simply compare the dictionaries using the equality operator. However, this is not feasible if it is possible that the server might be linked with more Slice definitions than the client. A more general solution is to iterate over the local checksums as demonstrated below:

C++

```
Ice::SliceChecksumDict serverChecksums = ...
Ice::SliceChecksumDict localChecksums = Ice::sliceChecksums();

for (Ice::SliceChecksumDict::const_iterator p = localChecksums.begin();
     p != localChecksums.end(); ++p) {

    Ice::SliceChecksumDict::const_iterator q = serverChecksums.find(p->first);
    if (q == serverChecksums.end()) {
        // No match found for type id!
    } else if (p->second != q->second) {
        // Checksum mismatch!
    }
}
```

In this example, the client first verifies that the server's dictionary contains an entry for each Slice type ID, and then it proceeds to compare the checksums.

See Also

- Slice Checksums
- slice2cpp Command-Line Options