C++ Logger Utility Classes

The Ice run time supplies a collection of utility classes that make use of the logger facility simpler and more convenient. Each of the `Logger` interface's four operations has a corresponding helper class:

C++11

```
namespace Ice
{
    class Print
    {
    public:
        Print(const std::shared_ptr<Logger>&);
        void flush();
        ...
    }

    class Trace
    {
    public:
        Trace(const std::shared_ptr<Logger>&, const std::string&);
        void flush();
        ...
    }

    class Warning
    {
    public:
        Warning(const std::shared_ptr<Logger>&);
        void flush();
        ...
    }

    class Error
    {
    public:
        Error(const std::shared_ptr<Logger>&);
        void flush();
        ...
    }
}
```

C++98

```
namespace Ice
{
    class Print
    {
    public:
        Print(const LoggerPtr&);
        void flush();
        ...
    }

    class Trace
    {
    public:
        Trace(const LoggerPtr&, const std::string&);
        void flush();
        ...
    }

    class Warning
    {
    public:
        Warning(const LoggerPtr&);
        void flush();
        ...
    }

    class Error
    {
    public:
        Error(const LoggerPtr&);
        void flush();
        ...
    }
}
```

The only notable difference among these classes is the extra argument to the `Trace` constructor; this argument represents the trace category.

To use one of the helper classes in your application, you simply instantiate it and compose your message:

C++

```
if(errorCondition)
{
    Error err(communicator->getLogger());
    err << "encountered error condition: " << errorCondition;
}
```

The Ice run time defines the necessary stream insertion operators so that you can treat an instance of a helper class as if it were a standard C++ output stream. When the helper object is destroyed, its destructor logs the message you have composed. If you want to log more than one message using the same helper object, invoke the `flush` method on the object to log what you have composed so far and reset the object for a new message.

The helper classes also supply insertion operators to simplify the task of logging an exception. The operators accept instances of `std::exception` (from which all Ice exceptions derive) and log the string returned by the `what` method. On some platforms, you can also enable the configuration property `Ice.PrintStackTraces`, which causes the helper classes to log the stack trace of the exception in addition to the value of `what`.

Back to Top ^

See Also

- Logger Facility

Previous