Tracing Program Execution

In this section:

Overview

Tracing is a very useful enhancement of Insure++ for C++ programmers. Because C++ is such a complicated language, programmers may never know which functions are being called or in which order. Some functions are called during initialization before the main program begins execution. Tracing provides the programmer with the ability to see how functions, constructors, destructors, and more are called as the program runs.

Insure++ prints a message at the entry to every function that includes the function name, filename, and line number of the command that called it. A typical line of output from tracing looks like this:

```
function_name filename, line_number
```

By default, the output is indented to show the proper depth of the trace.

Activating Tracing

Tracing is turned off by default. The easiest way to turn tracing on is to set the trace on value. This turns on tracing for the entire program. See Options Used by Insure++ for more information about this option.

Full traces require either the `-Zi` compiler switch (Windows) or `-g` flag (Unix) on your `insure` compile line. To get file names and line numbers in the trace output, you must use the `stack_internal on` option when compiling your program, which will slow your program down while every function call prints information.

This problem can be minimized by selectively turning on tracing during the execution of your program only in those sections of the code where you need it most. This can be done using the following Insure++ command in which `flag = 0` turns tracing off and `flag = 1` turns tracing on.

```
void _Insure_trace_enable(int flag)
```

There is an additional special Insure++ function that works with tracing. The following function may be used to add your own messages to the trace:

```
void _Insure_trace_annotate(int indent, char *format, ...)
```

In the above command, `indent = 0` means the string is placed in column zero, `indent = 1` means string will be indented at proper level, and `format` should be a normal `printf`-style format string.

Directing Tracing Output To A File

You can direct tracing output to a specific file by setting a `trace_file filename` value. You can set this value in the Windows GUI by choosing **Advanced tab> Advanced Configuration Settings for Insure++**. See Options Used by Insure++ for details.

When you use this option, Insure++ prints a message reminding you where the tracing data is being written. If you would like to eliminate these reminders, you can use the `trace_banner off` option. See Options Used by Insure++ for additional information.

Example

The following code can be found in the examples\cpp directory as the file trace.cpp:

```
/*
* File: trace.cpp
*/
int twice(int j) {
        return j*2;
}
class Object {
public:
        int i;
        Object() {
                i = 0;
        }
        Object(int j) {
                i = j;
        }
        operator int() { return twice(i); }
};


int main() {
        Object o;
        int i;


        i = 0;
        return i;
}
```

You will see the following output if you compile and link `trace.cpp` with the `-Zoi "stack_internal on"` option and run the executable with the `trace on` value set:

```
main                                        [called by non-insure code]
 Object::Object                 trace.cpp, 21
 Object::operator int        trace.cpp, 24
  twice                                        trace.cpp, 17
```

For more information about these and other options see Options Used by Insure++.