Exception Properties

This lesson explains the information that is automatically printed on the output when the program terminates due to an exception.

WE'LL COVER THE FOLLOWING

Exception properties

The information that is automatically printed on the output when the program terminates due to an exception is available as properties of exception objects as well. These properties are provided by the `Throwable` interface:

- `.file:` the source file where the exception was thrown from
- `.line:` the line number where the exception was thrown from
- `.msg:` the error message
- `.info:` the state of the program stack when the exception was thrown
- `.next:` the next collateral exception

We saw that finally blocks are executed when leaving scopes due to exceptions as well.

Collateral exception

Naturally, code in the finally blocks can throw exceptions as well. Exceptions that are thrown while leaving scopes due to an already thrown exception are called **collateral exceptions**. Both the main exception and the collateral

exceptions are elements of a linked list data structure, where every exception

object is accessible through the `.next` property of the previous exception object. The value of the `.next` property of the last exception is null.

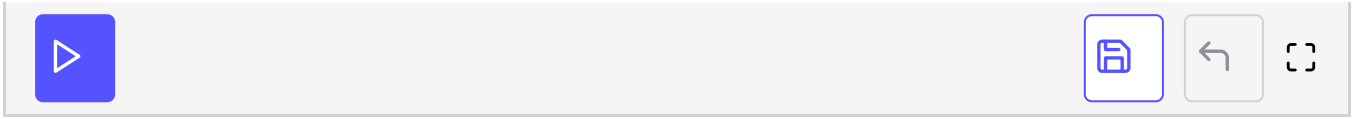
Example

There are three exceptions that are thrown in the example below: the main exception that is thrown in `foo()` and the two collateral exceptions that are thrown in the finally blocks of `foo()` and `bar()`. The program accesses the collateral exceptions through the `.next` properties.

Some of the concepts that are used in this program will be explained in later chapters. For example, the continuation condition of the for loop that consists solely of `exc` means as long as `exc` is not null.

```
module exceptions_9;

import std.stdio;

void foo() {
    try {
        throw new Exception("Exception thrown in foo");

    } finally {
        throw new Exception(
            "Exception thrown in foo's finally block");
    }
}

void bar() {
    try {
        foo();

    } finally {
        throw new Exception(
            "Exception thrown in bar's finally block");
    }
}

void main() {
    try {
        bar();

    } catch (Exception caughtException) {

        for (Throwable exc = caughtException;
             exc;    // ← Meaning: as long as exc is not 'null'
             exc = exc.next) {

            writefln("error message: %s", exc.msg);
            writefln("source file  : %s", exc.file);
            writefln("source line  : %s", exc.line);
```

```
            writeln();
        }


    }
}
```



Properties of exception

The next lesson will explain different types of errors and how to deal with those errors using exceptions.