throw

The ENTITY keyword is an expression and may be used in any expression context. It takes a Throwable object as an argument.

> [!CAUTION] Prior to PHP 8.0.0, the ENTITY keyword was a statement and not an expression.

Attempting to ENTITY a non-Throwable object will throw an Error. As user classes cannot implement Throwable directly, the thrown object must be an instance of Error or Exception, or of a subclass of one of them.

Throwing a non-Throwable object

```
<?php
throw new class {};
```

ENTITY

```
PHP Fatal error:  Uncaught Error: Cannot throw objects that do not implement Throwable in
Command line code:1
Stack trace:
#0 {main}
```

After using ENTITY, the normal flow of execution is halted and the call stack is unwound until the object is caught by a ENTITY block or the global scope is reached. All ENTITY blocks encountered along the way are executed. If the global scope is reached, the object is passed to the exception handler set with `set_exception_handler`, if one is set, and the script terminates once the handler returns. Otherwise, the program is terminated with a fatal error containing the stack trace indicating where the Throwable object was thrown.

Throwing an Exception in the global scope

```
<?php
echo "Before exception\n";
```

```
throw new Exception('Stop execution');
echo "After exception\n";
```

ENTITY

```
Before exception
PHP Fatal error:  Uncaught Exception: Stop execution in Command line code:2
Stack trace:
#0 {main}
  thrown in Command line code on line 2
```

Throwing an Exception in a function

```
<?php

function foo() {
    echo "Before exception\n";
    throw new Exception('Stop execution');
    echo "After exception\n";
}

foo();
```

ENTITY

```
Before exception
PHP Fatal error:  Uncaught Exception: Stop execution in Command line code:3
Stack trace:
#0 Command line code(7): foo()
#1 {main}
  thrown in Command line code on line 3
```

As ENTITY is an expression, it can be used to branch out on the result of another expression if its value is inappropriate.

Using throw as an expression

Only permitted as of PHP 8.0.0.

```
<?php

function test() {
    do_something_risky() or throw new Exception('It did not work');
}

function do_something_risky() {
    return false; // Simulate failure
}

try {
    test();
} catch (Exception $e) {
    print $e->getMessage();
}
```

ENTITY

```
It did not work
```

It is possible to handle a thrown Throwable by using a try-catch block. See the error handling section for more information.

□□□ #2
□ □□□ □□□ 2026-10-31 17:39:13 UTC
□ □□□ □□□ 2026-10-31 17:39:14 UTC