getcurrent

Closure::getCurrent

Returns the currently executing closure

ENTITY

public

static

Closure

Closure::getCurrent

Returns the currently executing closure. This method is primarily useful for implementing recursive closures without needing to capture a reference to the closure variable using the `use` keyword.

This method must be called from within a closure; calling it outside of a closure context will result in `Error: Current function is not a closure.`

ENTITY ENTITY

ENTITY

Returns the currently executing `Closure` instance.

ENTITY

Throws an Error if called outside of a closure context.

ENTITY

Closure::getCurrent example

Using Closure::getCurrent to implement a recursive Fibonacci function:

```
<?php
$fibonacci = function (int $n) {
    if ($n === 0 || $n === 1) {
        return $n;
    }

    $fn = Closure::getCurrent();
    return $fn($n - 1) + $fn($n - 2);
};

echo $fibonacci(10); // Outputs: 55
?>
```

Comparison with traditional approach

Prior to PHP 8.5, implementing recursive closures required capturing a reference to the closure variable using the `use` keyword:

```
<?php
// Traditional approach (still works in PHP 8.5)
$fibonacci = function (int $n) use (&$fibonacci) {
    if ($n === 0) return 0;
    if ($n === 1) return 1;
    return $fibonacci($n - 1) + $fibonacci($n - 2);
};

echo $fibonacci(10); // Outputs: 55
?>
```

The Closure::getCurrent approach eliminates the need to declare the variable with a reference in the `use` clause, making the code cleaner and less error-prone.

□□□ #2
□ □□□ □□□ 2026-10-31 17:42:44 UTC
□ □□□ □□□ 2026-10-31 17:42:45 UTC