

Casting

Recursively interpret array values through `ValueParser` with optional numeric preservation

Anonymous · 26/10/2026

ArrayCaster

`ArrayCaster` traverses array leaves and delegates each value directly to `Derafu\Value\ValueParser::parse()`. Its options and defaults match the parser:

```
ArrayCaster::cast(  
    array $array,  
    bool $preserveNumericFormat = false,  
    bool $preserveNumericPrecision = false,  
): array
```

The returned array keeps its original keys and nesting. The supplied array is not modified. Empty nested arrays remain empty arrays. Scalar interpretation, Stringable conversion and unsupported values follow [ValueParser](#).

Recursive Interpretation

```
use Derafu\Array\ArrayCaster;  
  
$input = [  
    'quantity' => '12.0',  
    'message' => ' text ',  
    'blank' => ' ',  
    'empty' => 'empty',  
    'nothing' => '(null)',  
    'nested' => [3 => ['enabled' => 'off', 'zero' => '0']],  
];  
  
$result = ArrayCaster::cast($input);  
// [  
//     'quantity' => 12,  
//     'message' => ' text ',  
//     'blank' => false,  
//     'empty' => '',  
//     'nothing' => null,  
//     'nested' => [3 => ['enabled' => false, 'zero' => 0]],  
// ]
```

There is no trimming or empty-value substitution before the parser. Unrecognized text keeps its spaces, while whitespace-only text becomes `false` through `ValueCaster`'s boolean conversion. Integers are preferred when exact conversion is possible; numeric `'0'` and `'1'` therefore become integers rather than booleans.

Native numbers, booleans, nulls, resources and objects without `Stringable` remain unchanged. `Stringable` leaves are converted to text and interpreted using the same options. A failing `Stringable` conversion propagates `Value`'s translatable `ValueConversionException` with its original cause.

Preserving Numeric Format

Pass `preserveNumericFormat: true` to keep numeric text when converting the resulting number back to string would alter its original spelling:

```
use Derafu\Array\ArrayCaster;

ArrayCaster::cast(
    ['a' => '12', 'b' => '19.90', 'c' => '007', 'd' => '1e3', 'e' => ' 12 '],
    preserveNumericFormat: true,
);
// ['a' => 12, 'b' => '19.90', 'c' => '007', 'd' => '1e3', 'e' => ' 12 ']
```

Exterior whitespace and decimal trailing zeros are part of the original spelling. This option concerns numeric text; special literals and boolean words are still interpreted normally.

Preserving Numeric Precision

Pass `preserveNumericPrecision: true` to retain numeric text when the parser detects lost significant decimal digits or a non-finite result:

```
use Derafu\Array\ArrayCaster;

ArrayCaster::cast(['large' => '1e400']);
// ['large' => INF]

ArrayCaster::cast(
    ['large' => '1e400', 'small' => '1e-400', 'fraction' => '1.00000000000000000001'],
    preserveNumericPrecision: true,
);
// ['large' => '1e400', 'small' => '1e-400', 'fraction' => '1.00000000000000000001']
```

Both options may be enabled together. Their exact guarantees, including the distinction between significant decimal digits and exact binary representation, are defined by [ValueParser](#). `ArrayCaster`

implements no separate numeric grammar or preservation rules.

Reading and Conversion

Use `ArrayReader`'s strict getters to validate existing PHP types, or `getAsXXX()/requireAsXXX()` to convert individual fields. Use `ArrayCaster` when an entire array should be traversed. Recursive interpretation does not declare a required field schema or choose application defaults.

Last updated on 26/10/2026