

Cheatsheet: Creating mocks in PHPUnit

Available in
test cases

Name of the
class to mock
(= to subclass)

names of the
methods to mock
(empty = all,
null = none)

parameters for
the constructor

name of the
created class
(empty = auto)

whether to call
the original
constructor

```
$mock = $this->getMockBuilder(CoffeeCup::class),  
->setMethods(['fill', 'stir'])  
->setConstructorArgs([$size, $color]),  
->setMockClassName('MockedCup')  
->disableOriginalConstructor()  
->getMock();  
or: ->getMockForAbstractClass();
```

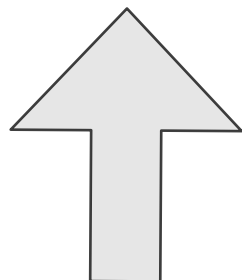
all calls except
for the first
and last one
can be omitted

Why mock a method?

- To "disable" a method (to not write to the DB, or to not launch a cruise missile) and return null.
- To have the method return a particular return value.
- To test that the method gets called in a certain way.

Cheatsheet: Using mocks in PHPUnit

```
$mock->expects(self::once())  
->method('fill')  
->with($coffee, 200)  
->will(  
    self::returnValue(true)  
);
```



any of the last
two calls can be
omitted

how often:

never()
once()
any()
atLeastOnce()
atLeast(3)
exactly(4)

or which call:

at(0) for the first call
at(2) for the third call

name of the method,
must be a mocked
method

required parameters
or self::anything(),
omit for never()

return value