Creating Fake Objects

Dror Helper

@dhelper http://blog.drorhelper.com

Module Overview

Why we need fake objects

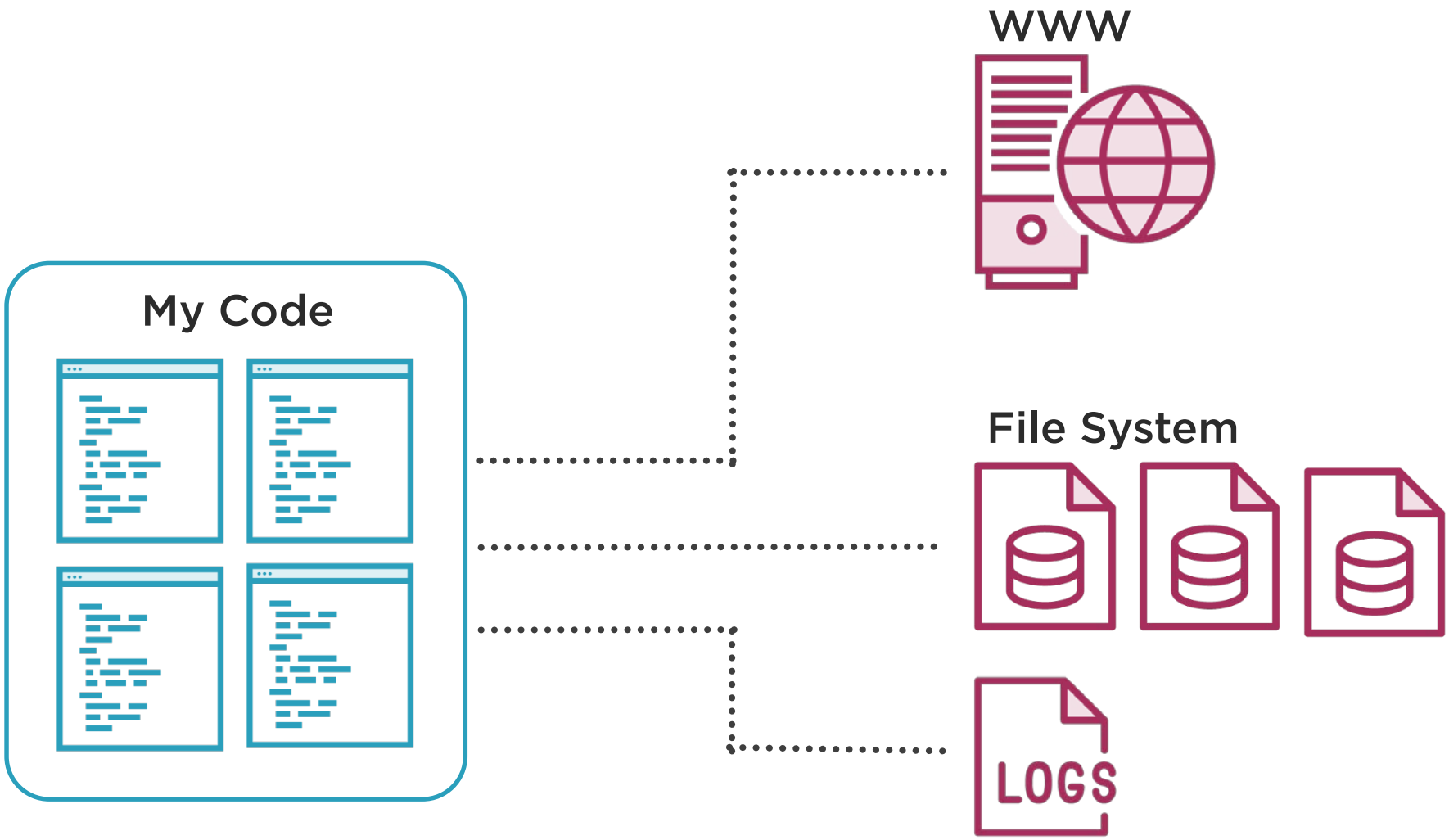
Mocks, Stubs, Fakes...

Writing Fake Objects

Faking with Google Mocks

Harnessing Dependency Injection

Real Code Has Dependencies



Two Types of Tests

Unit Tests

Fast

Isolated

Integration Tests

Has dependencies

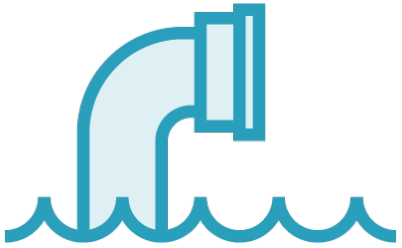
Require environment setup

Challenges of Integration Tests

Difficult to initialize

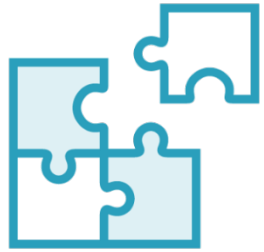
Hard to test cases

Slow tests



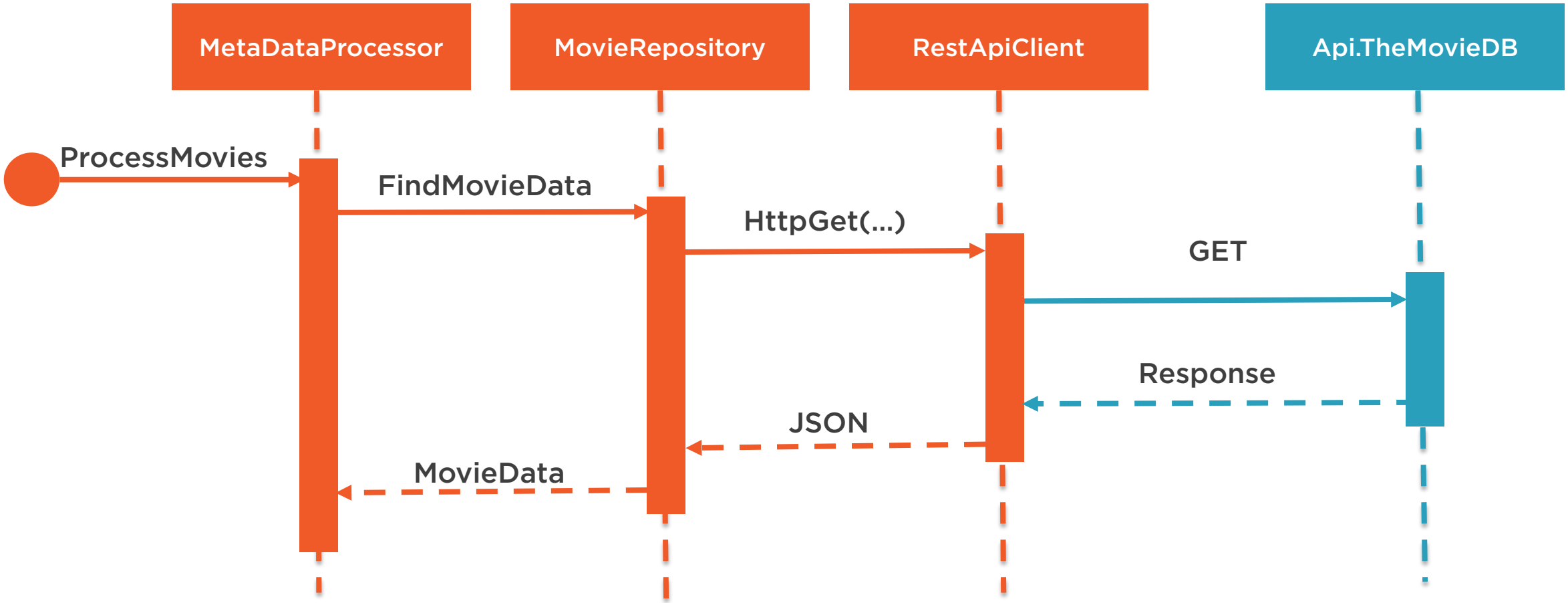
Difficult to verify test results

Side Effects

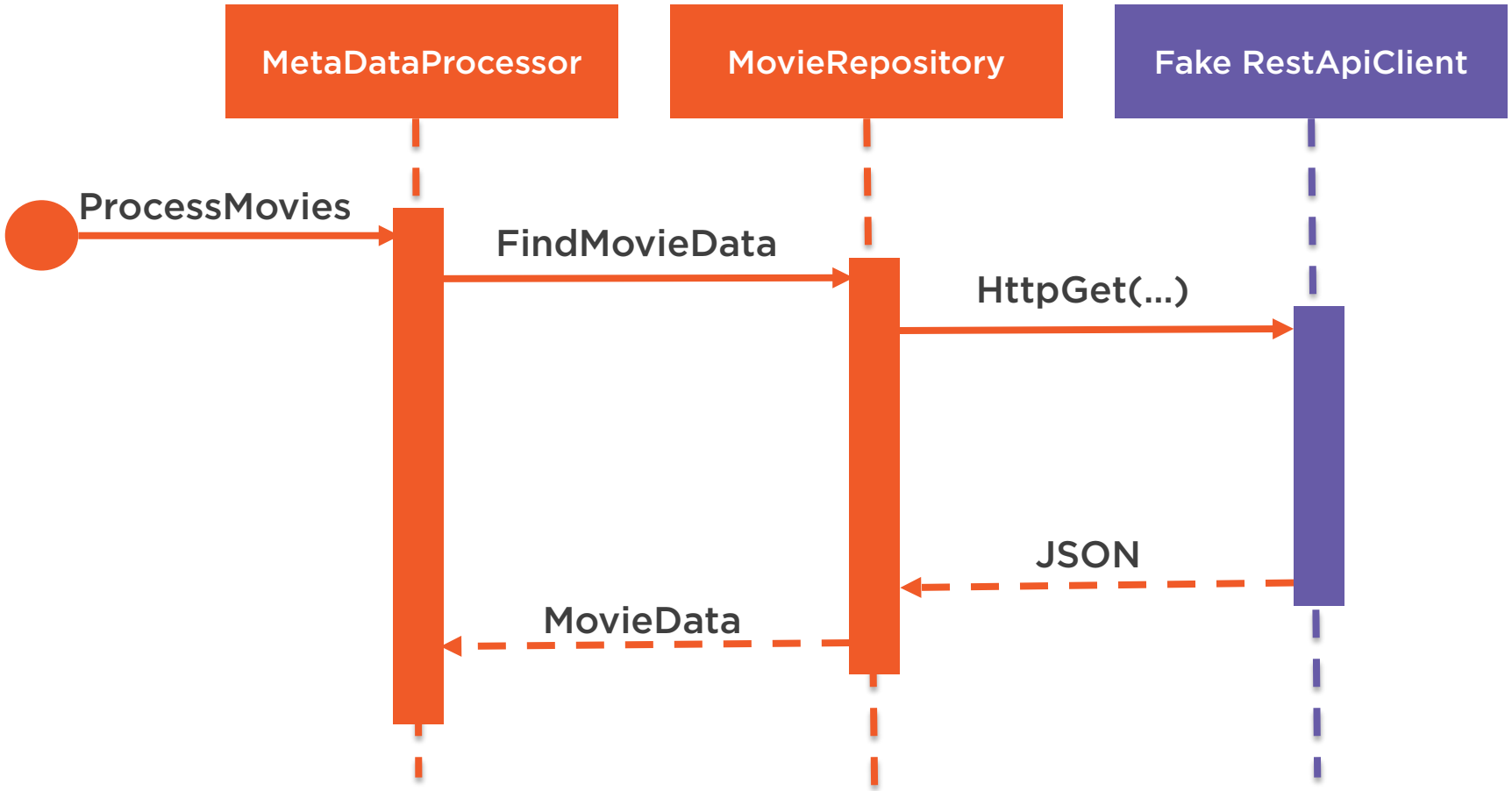


Dependent on environment

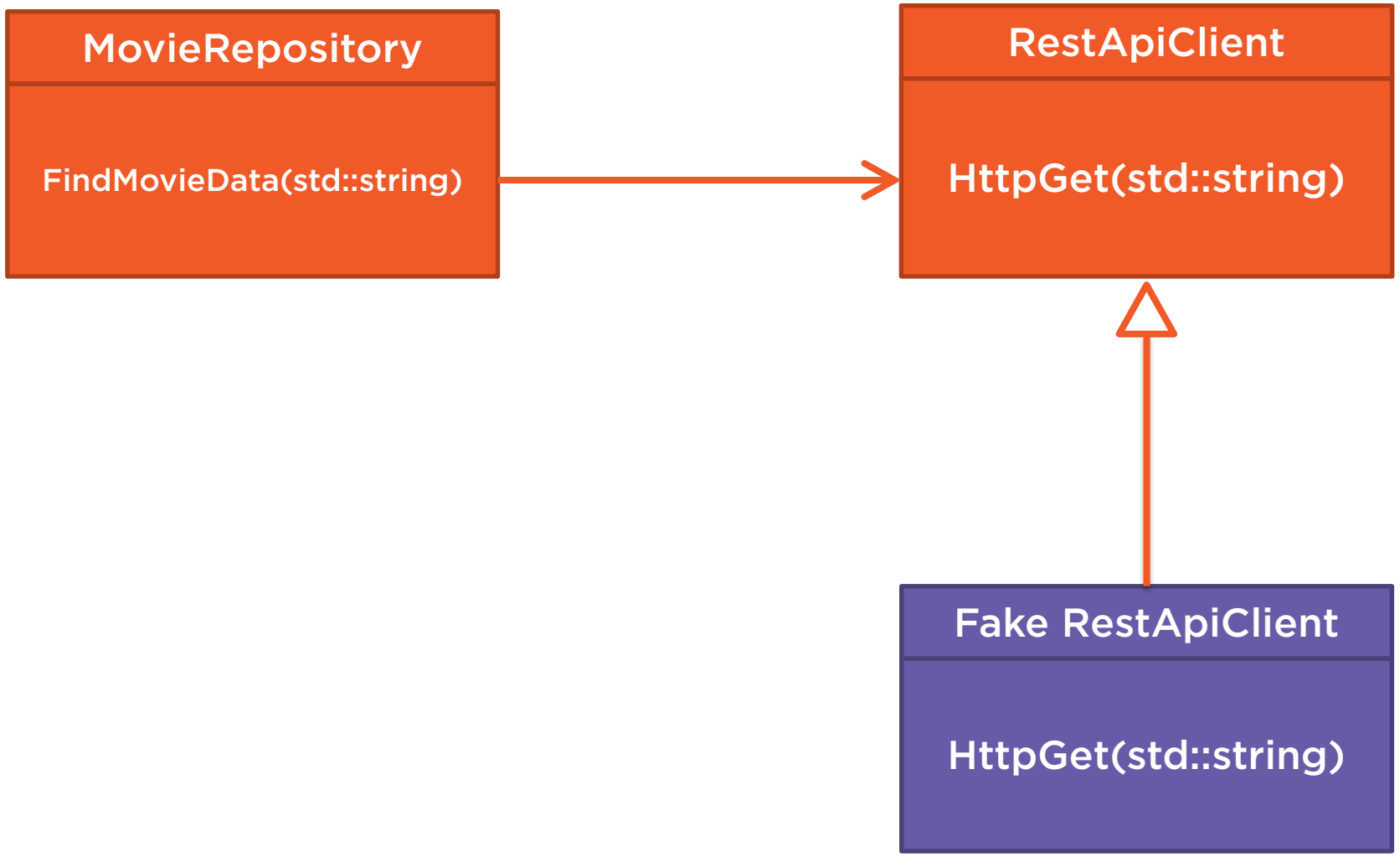
Test Isolation → Removing Dependencies



Test Isolation → Removing Dependencies



Writing Hand Rolled Fake Objects



Never Write Your Own Fake Objects

Maintainability

Complexity

Logic duplication

Error prone

Google's C++ Mocking Framework

Or GMock for short,

Is an open sourced, free and extendible library for creating mock classes and using them

The difference between Mocks, Stubs and Fakes

Reminder: Getting Started with GMock

```
#include "gtest/gtest.h"
#include "gmock/gmock.h"

int main(int argc, char** argv)
{
    ::testing::InitGoogleMock(&argc, argv);
    return RUN_ALL_TESTS();
}
```

Creating Fake Objects

```
class FakeRestApiClient : public RestApiClient
{
public:
    MOCK_METHOD2(HttpPost, void(string&, string&));
    MOCK_METHOD1(HttpGet, string(string&));
    MOCK_METHOD2(HttpPut, void(string&, string&));
    MOCK_METHOD1(HttpDelete, void(string&));
};
```

Faking Methods

Real class must have a virtual destructor

Faked methods must be *virtual*

Use the right macro

- MOCK_METHODx
- MOCK_CONST_METHODx

```
gmock_gen.py header-file.h [class name]
```

Google Mock Class Generator

Found at: *googlemock\scripts\generator*

Requires Python 2.4 installed

Output result to *stdout*

Faking Class Templates

```
template <typename T>
class StackInterface
{
     virtual int GetSize() const =0;
     virtual int Push(const T& x) = 0;
};
```

```
template <typename T>
class FakeStack : public StackInterface<T>
{
     MOCK_CONST_METHOD0_T(GetSize, int());
     MOCK_METHOD1_T(Push, void(const T& x));
};
```

```
class File : public FileInterface {
    virtual bool Open(const char* path, const char* mode) {
        return OpenFile(path, mode);
    }
}
```

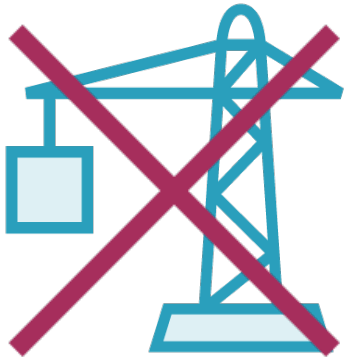
Static and Free Functions

No inheritance → Cannot work

Instead wrap in a class

Dependency Injection

Pass dependency to
the object that uses it



Creating
dependencies inside
class is prohibited

Client code
do not change due to
dependency change

Dependency Injection Technics

Constructor Injection

Method Injection

Setter Injection

Other

Prepare Your Code For Faking

Make All methods virtual (incl. d'tor)

Use pure virtual base classes

Wrap static and free methods

Use Dependency Injection

Summary

Faking dependencies

- Writing fake objects
- Using Gmock

Dependency Injection