Working With Legacy Code

@maciejoczko

#tddberlin

What is Legacy Code?

Code is you didn't write this morning (yet).

What is Legacy Code?

Code is legacy code as soon as it's written.

Michael Feathers, Working Effectively with Legacy Code

What is Legacy Code?

Code without tests*

Inherited code

Poorly designed code

Too complicated code

Illegible code

Testing scenario #1 Task

How to approach it?
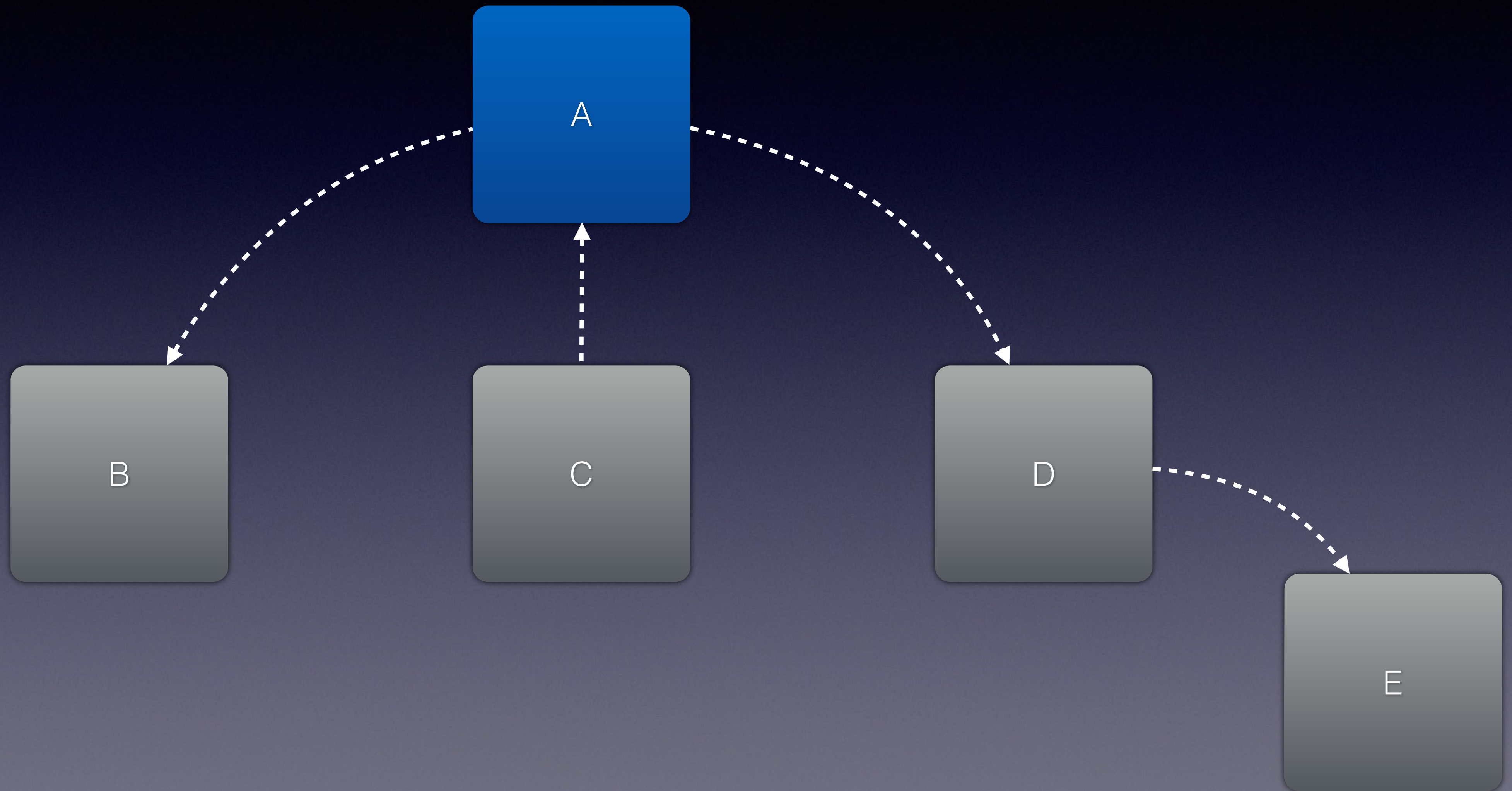
Identify change points

Find an inflection point

Cover the inflection point

Make changes

Refactor the covered code

inflection point

inflection point

Violation of Single Responsibility Principle

Violation of Open-Closed Principle

Covering inflection point

Break external dependencies

Covering inflection point

Break external dependencies

Break internal dependencies

Covering inflection point

Break external dependencies

Break internal dependencies

Write tests

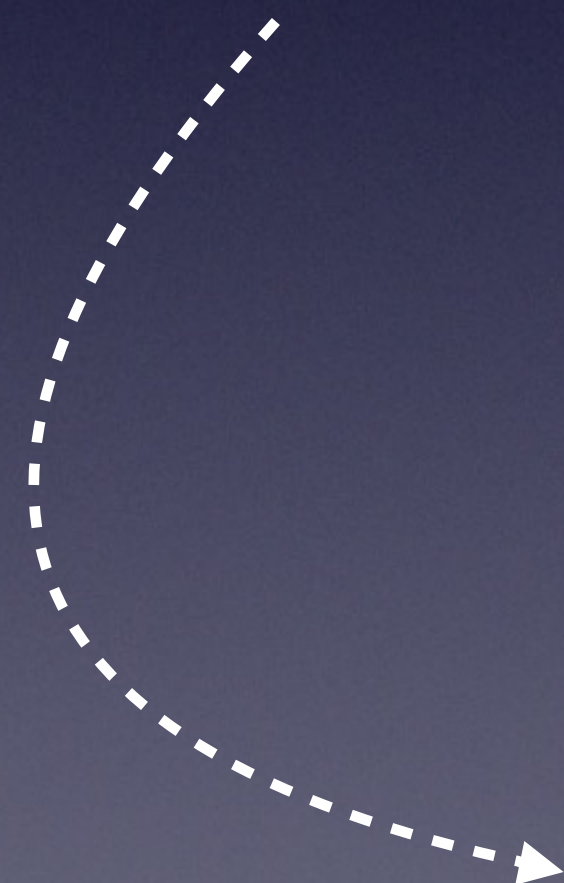# Covering inflection point

Break external dependencies

Break internal dependencies

Write tests

Break **external** dependencies

```objc
- (NSUInteger)calculatePrice {
    ...
    StockAnalyzer *analyzer = [[StockAnalyzer alloc] init];
    ...
}
```

```objc
- (id)initWithAnalyzer:(id <AnalyzerProtocol>)analyzer {
    self = [super init];
    if (self) {
        _analyzer = analyzer;
    }
    return self;
}
```
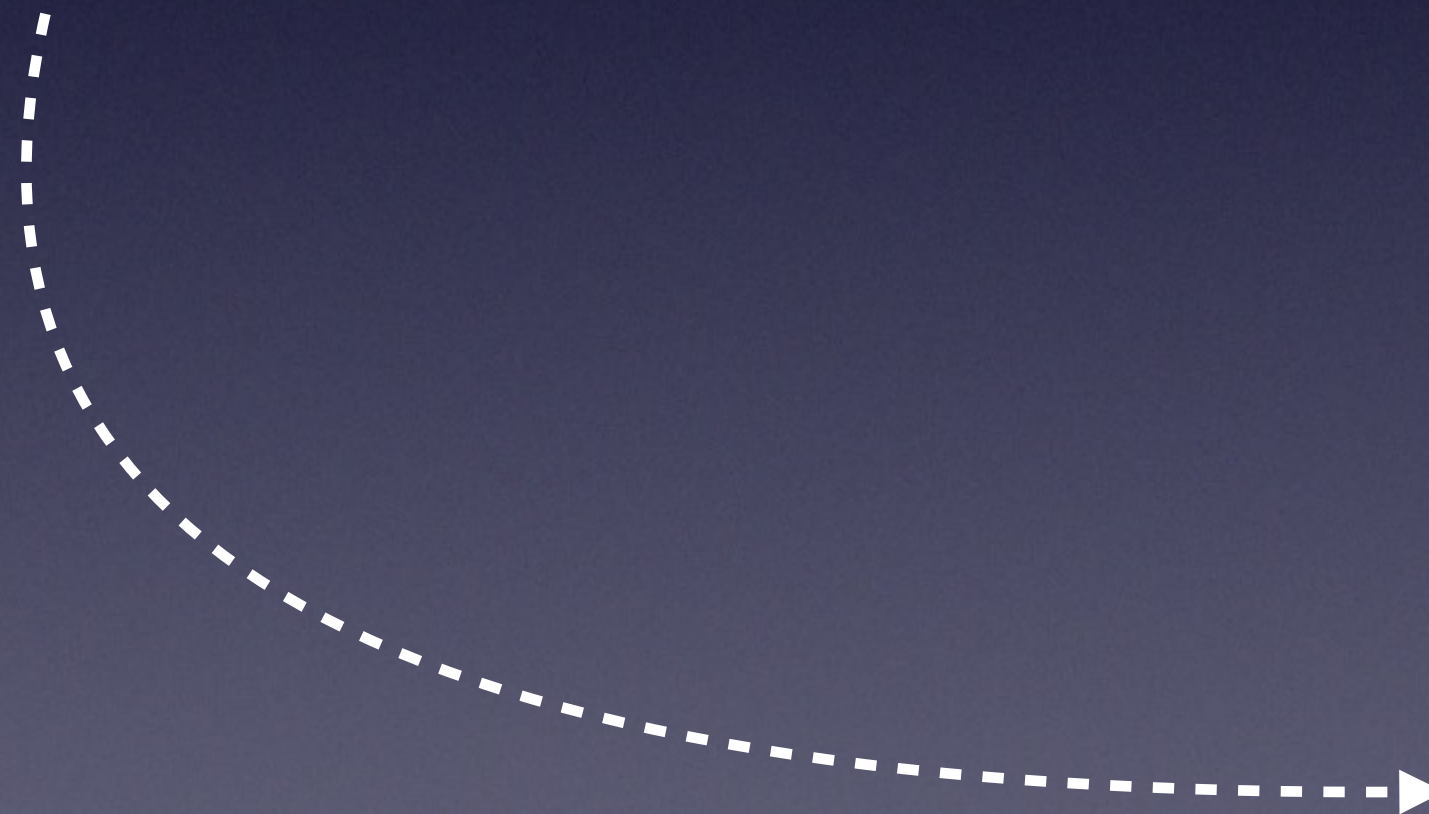
Covering inflection point

Break external dependencies

Break internal dependencies

Write tests

Break **internal** dependencies

```objc
- (CGSize)calculateSize {
    ...
    UIScreen *screen = [UIScreen mainScreen];
    ...
}
```

```objc
- (CGSize)calculateSize {

    ...
    UIScreen *screen = [self screen];
    ...
}

- (UIScreen *)screen {
    return [UIScreen mainScreen];
}
```

Covering inflection point

Break external dependencies

Break internal dependencies

Write tests

Write tests

Make changes

Testing scenario #2 Task

When to Refactor?

Too wide class responsibility

Duplicated code

Not readable code

Too many method arguments (more than 2)

Method has more than 20 lines

Class has more than 150-200 lines

Keep in mind

Composition over inheritance

Avoid singletons

Isolate dependencies

Inject dependencies

Thanks!

@maciejoczko

#tddberlin