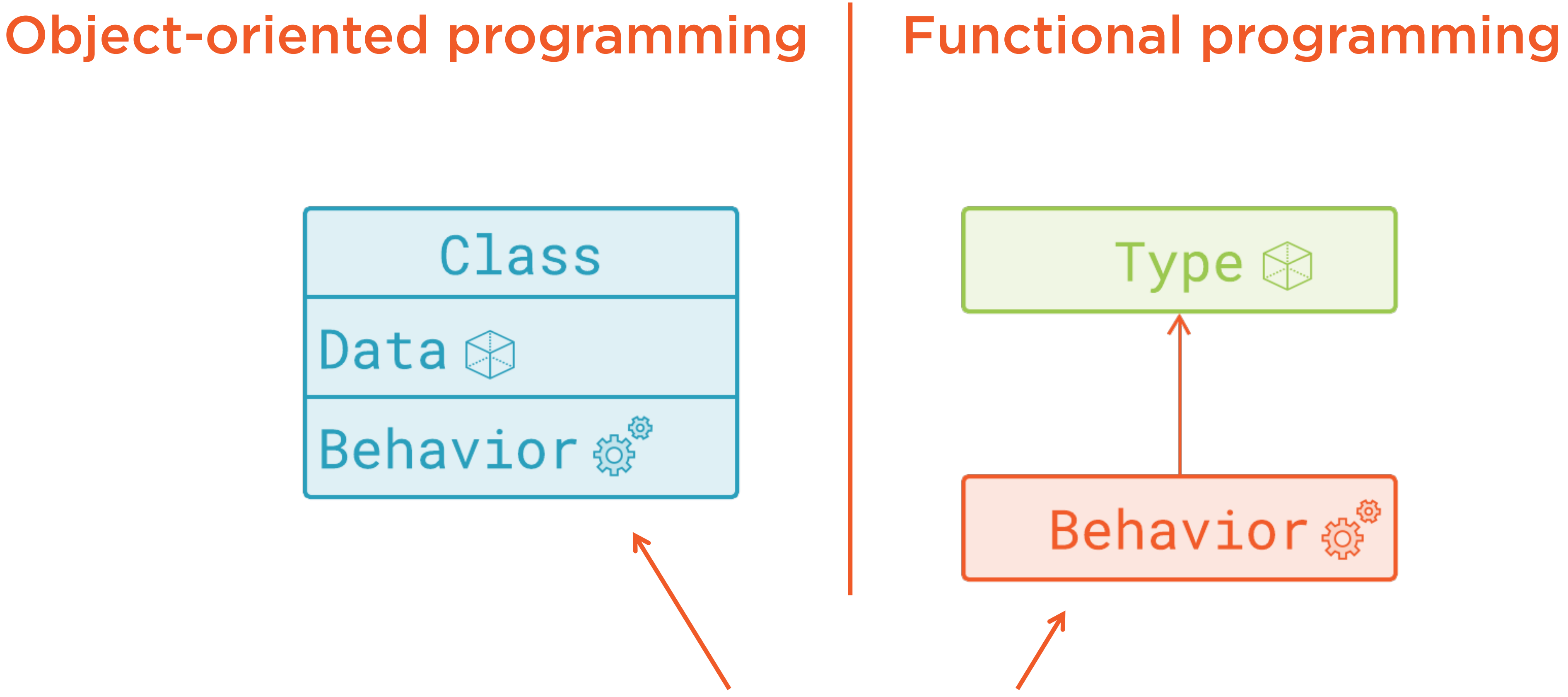
Function Composition with Object Model

Zoran Horvat

CEO AT CODING HELMET

@zoranh75 http://csharpmentor.com

Function Composition



Function composition applies to both coding styles

Function Composition

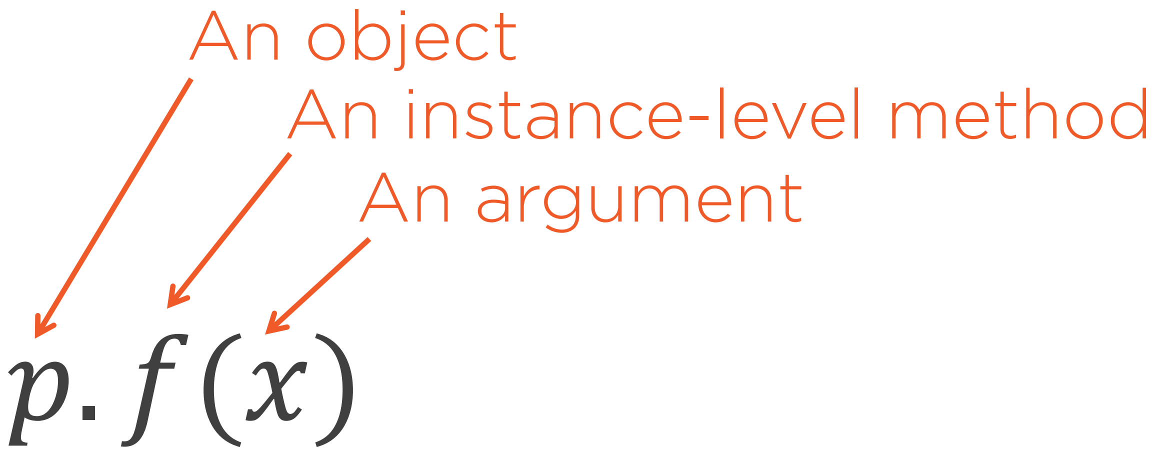
$$(f \circ g)(x)$$

Function composition operator

Function Composition

$$(f^{\circ}g)(x) = g(f(x))$$

Function Composition



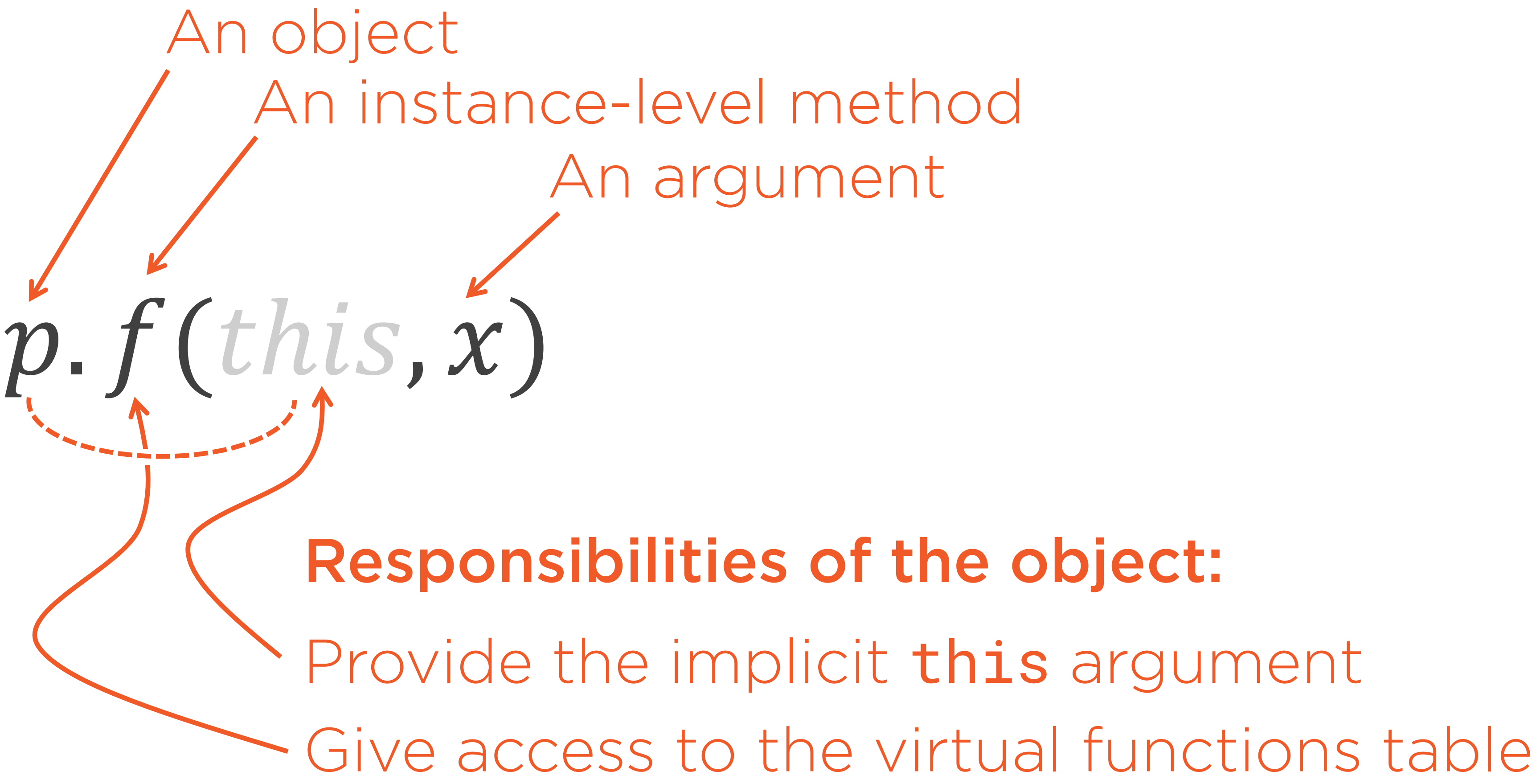
An object

An instance-level method

An argument

$$p.f(x)$$

Function Composition



An object

An instance-level method

An argument

$$p.f(this, x)$$

Responsibilities of the object:

Provide the implicit **`this`** argument

Give access to the virtual functions table

Function Composition

$$y = p.f(x)$$

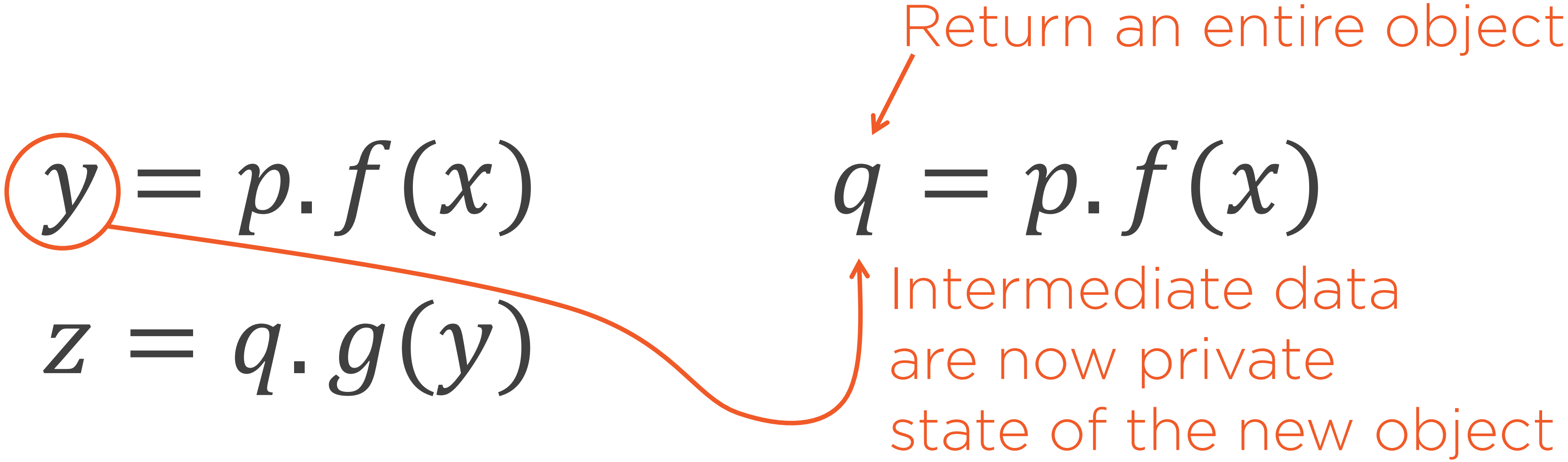
$$z = q.g(y)$$

$$z = q.g(p.f(x))$$

```
finalResult = serviceManager
  .FetchLongAndDetailed(
    intermediaryWithPowerBenefits
      .DecrementBalanceFor(
        youDontSeeMe));
```

In practice,
often looks
more like this

Function Composition



$$y = p.f(x)$$

$$z = q.g(y)$$

$$q = p.f(x)$$

Return an entire object

Intermediate data
are now private
state of the new object

Function Composition

$$y = p.f(x)$$

$$z = q.g(y)$$

$$q = p.f(x)$$

$$z = q.g(\,)$$

Only the **`this`** reference remains as the implicit argument

Function Composition

$$z = p.f(x).g();$$

```
public static class Extensions
{
  public static Q f(this object x) =>
    new P().f(x);
}
```

$$z = x.f().g();$$

Function Composition

Fluent interface produces this:

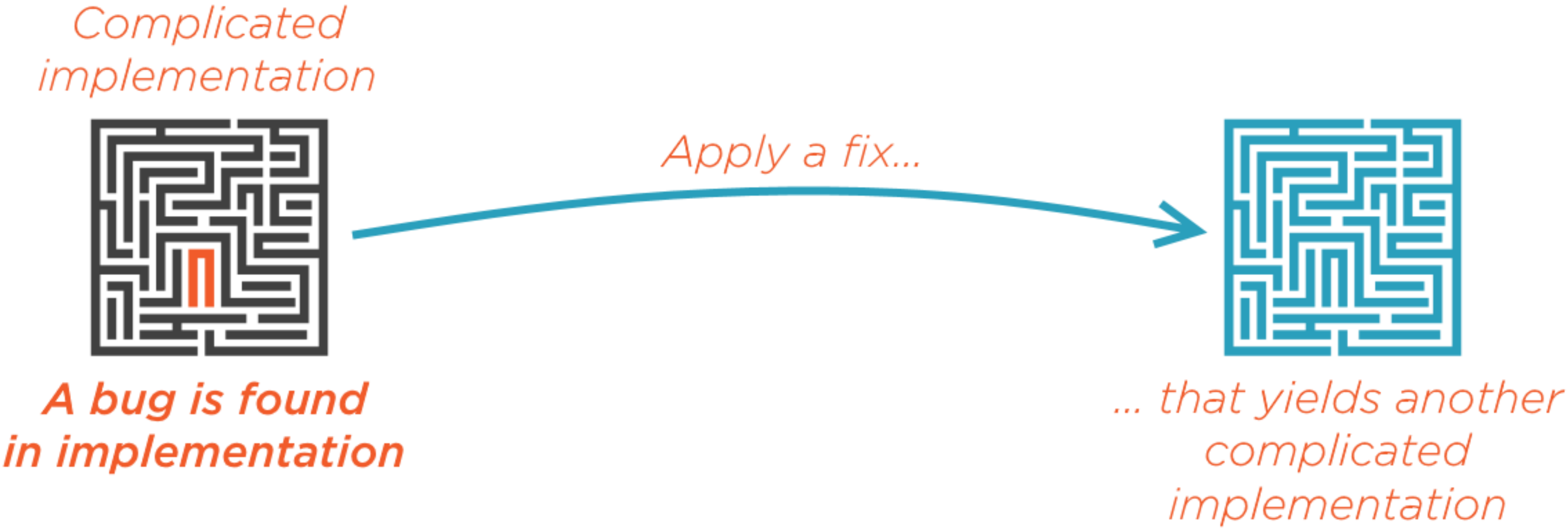
```
finalResult =
  youDontSeeMe
    .DecrementBalance()
    .FetchLongAndDetailed();
```

Instead of this:

```
finalResult = serviceManager
  .FetchLongAndDetailed(
    intermediaryWithPowerBenefits
      .DecrementBalanceFor(
        youDontSeeMe));
```

$$z = x.f().g();$$

To Fix or to Redesign?



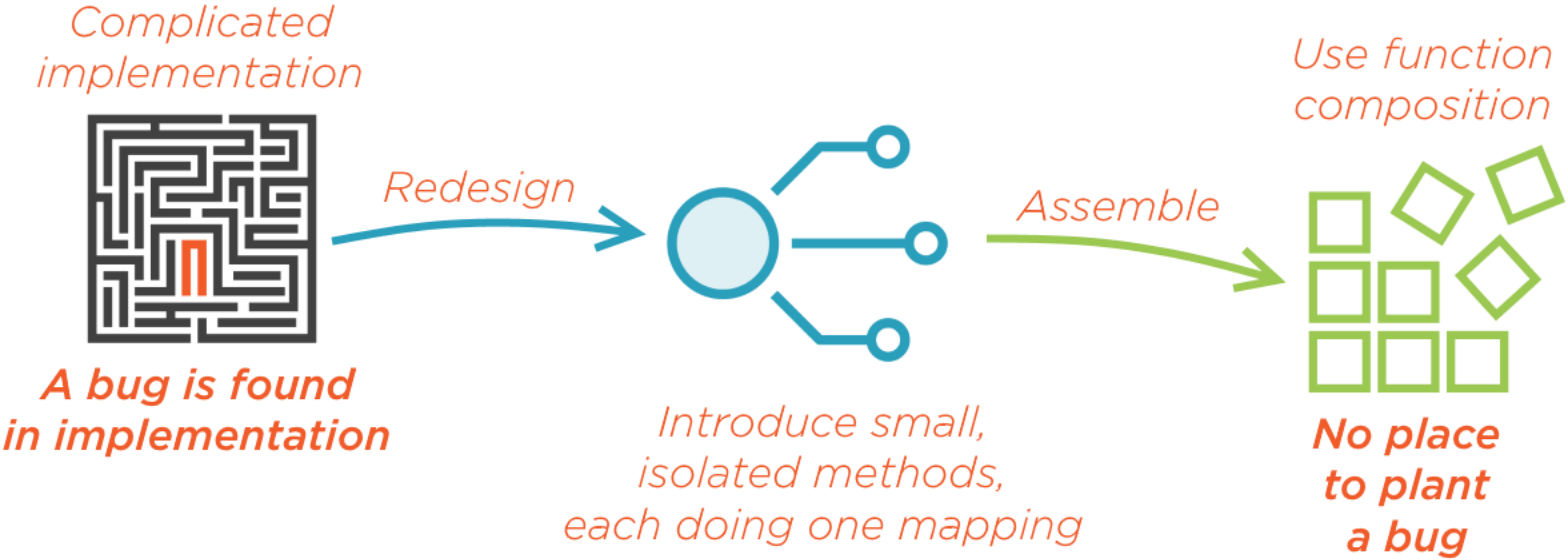
Complicated implementation

A bug is found in implementation

Apply a fix...

... that yields another complicated implementation

To Fix or to Redesign?



Complicated implementation

A bug is found in implementation

Redesign

Introduce small, isolated methods, each doing one mapping

Assemble

Use function composition

No place to plant a bug

Summary

Function composition

- Passing output of one function to an input of another function
- Looks the same as one large function
- Composable functions are easy to implement

Function composition in C#

- One function returns an object
- That object exposes the next function
- Second function receives the implicit `this` reference
- Method chaining/fluent interface design

Next module:
Railway-oriented Programming