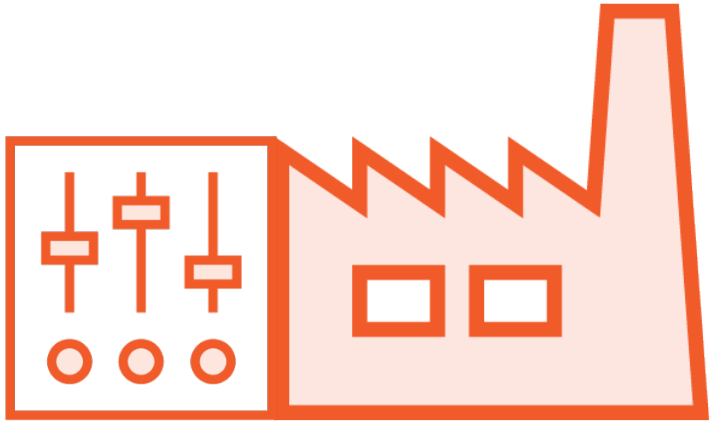
Evolving an Object Mother Into an Object Builder

Mel Grubb
DEVELOPER

@melgrubb melgrubb.com github.com/melgrubb

The Object Builder



```
var orders = context.Orders
    .Where(x => x.UserId == userId && !x.IsDeleted)
    .Include(x => x.OrderItems)
    .ThenInclude(x => x.Product)
    .OrderByDescending(x => x.Date)
    .ThenBy(x => x.Status)
    .ToList();
```

Fluent Interfaces

Chain together multiple calls to one or more classes

Designed to read like a sentence

Aids in discoverability

```
var country = new CountryBuilder
    .WithAbbreviation(“TC”)
    .WithName(“Test Country”)
    .WithoutDescription()
    .Build();
```

Fluent Builders

Builds up a description of the object you want

Per-property “With” methods

Build method

Demo

From Object Mother to Object Builder

```
var country = CountryBuilder
    .Simple()
    .WithAbbreviation("TC")
    .WithName("Test Country")
    .WithoutDescription()
    .Build();
```

Factory Methods

Pre-defined or "canned" Builders

Allow for greater re-use of known objects or states

Can be further modified before finally calling Build

Demo

CountryBuilder

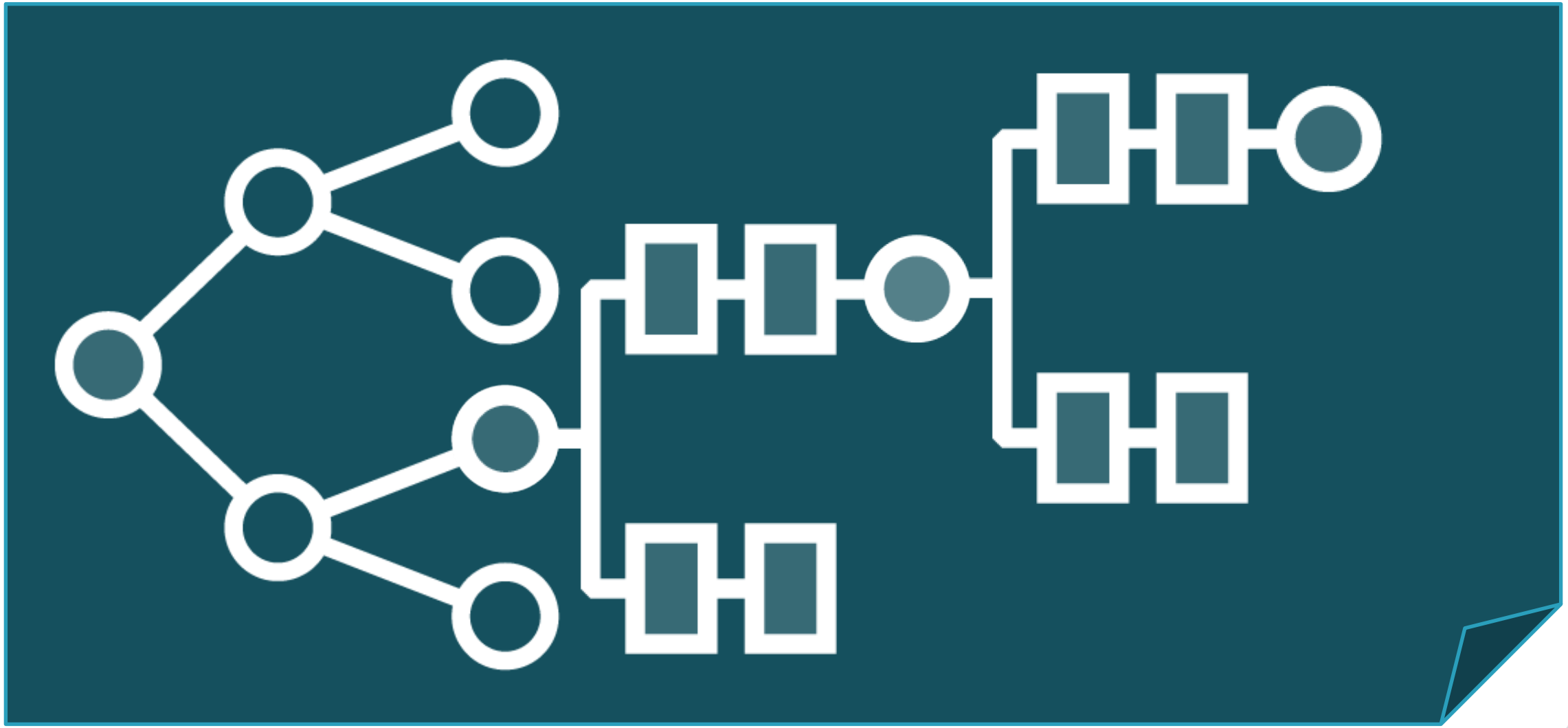
```
// CountryMother

protected override void Given()
{
    base.Given();

    _model = CountryMother

        .Typical();

    _model.FirstName = "Bob";

    _model.LastName = "Smith";

}
```

```
// CountryBuilder

protected override void Given()
{
    base.Given();

    _model = CountryBuilder

        .Simple()

        .WithFirstName(“Bob“)

        .WithLastName(“Smith“)

        .Build();

}
```

Demo

StateBuilder

A Builder Is a Blueprint



Demo

Make a plan

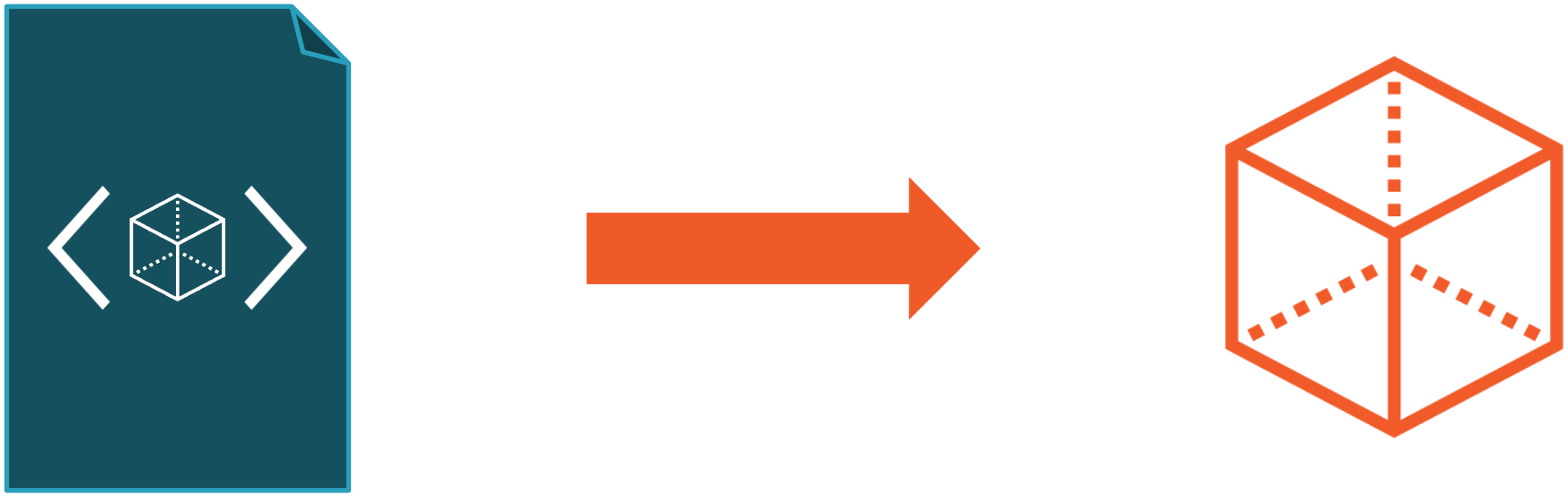
Lazy is Good

Defered object creation

Placeholder for a real value

Contains a function to get the value when needed

System.Lazy<T>



Demo

Lazy Builders

Summary

Created Simple Object Builders

Made them more flexible

Deferred execution as long as possible

Up Next

Generating the Builders