

Module 03: "Builder"



Agenda

- ▶ Introductory Example: Creating Pizzas
- ▶ Challenges
- ▶ Implementing the Builder Pattern
- ▶ Pattern: Builder
- ▶ Overview of Builder Pattern
- ▶ Fluent APIs



Introductory Example: Creating Pizzas

```
class Pizza
{
    public CrustKind Crust { get; set; }
    public bool HasSauce { get; set; }
    public IEnumerable<ToppingKind> Toppings { get; set; }
    public CheeseKind? Cheese { get; set; }
    public bool Oregano { get; set; }
}
```

```
Pizza hawaii = new Pizza
{
    Crust = CrustKind.Classic, HasSauce = true, Oregano = true,
    Cheese = CheeseKind.Regular,
    Toppings = new List<ToppingKind>
        { ToppingKind.Ham, ToppingKind.Pineapple }
};
```

Challenges

- ▶ Need to create the same configuration of object whenever needed
- ▶ Might end up with partially-defined objects
- ▶ Cannot define sequencing constraints

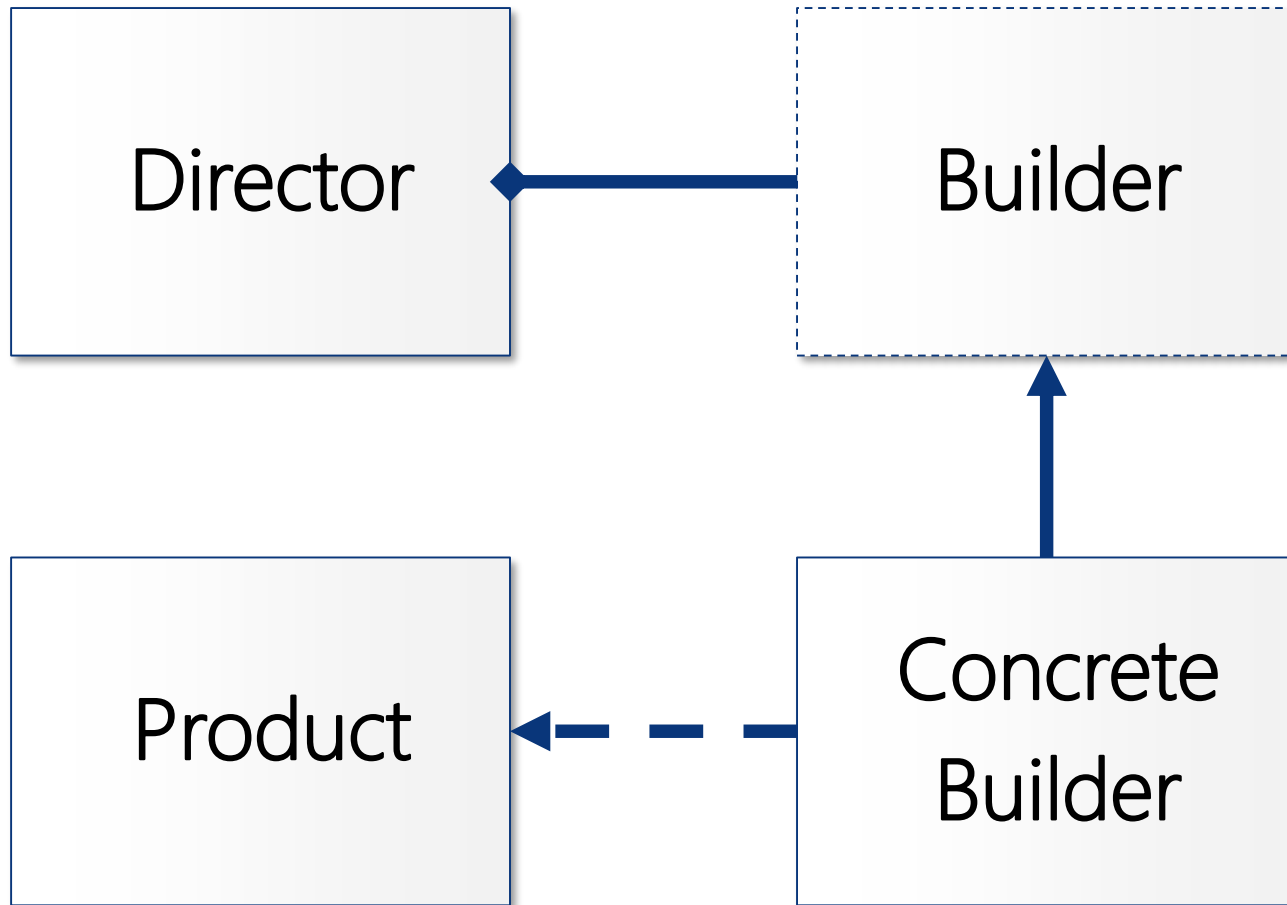


Pattern: Builder

- ▶ *Separate the construction of a complex object from its representation so that the same construction process can create different representations.*
- ▶ Outline
 - Construct complex objects in individual, reusable steps in the correct order.
 - Separate data from construction logic
 - reuse the construction logic
 - Construction data is handled by the Builder
 - Construction logic is handled by a Director
- ▶ Origin: Gang of Four



Overview of Builder Pattern



Overview of Builder Pattern

- ▶ Director
 - Knows the steps and the sequencing of the construction process
 - Just needs the concrete data pieces
- ▶ Builder
 - Abstract class or interface
 - Knows the data pieces
 - Keeps the Product
- ▶ Concrete Builder
 - Concrete class implementing the general Builder specification
 - Knows the concrete data pieces for a specific variation of object
- ▶ Product
 - Composite or complex objects
 - A single type of Product
 - but the data of the product object varies



Fluent APIs

- ▶ Fluent APIs configure the Builder instances in a “fluent” fashion
 - Flows like natural, spoken language
 - See Lab 03.1

```
FluentPizzaBuilder builder = new FluentPizzaBuilder();  
Pizza hawaii = builder  
    .Begin()  
    .WithCrust(CrustKind.Classic)  
    .Sauce  
    .AddTopping(ToppingKind.Ham)  
    .AddTopping(ToppingKind.Pineapple)  
    .AddCheese()  
    .Oregano  
    .Build();
```