

# Module 09: "Composite"



# Agenda

- ▶ Introductory Example: Wincuburger Combos
- ▶ Challenges
- ▶ Implementing the Composite Pattern
- ▶ Pattern: Composite
- ▶ Overview of Composite Pattern



# Introductory Example: Wincuburger Combos

```
class SingleItem
{
    public string Description { get; set; }
    public decimal Price { get; set; }
    public override string ToString() => $"{Description} [DKK {Price}]";
}
```

```
SingleItem burger = new SingleItem
{ Description = "Mic Bag Burger", Price = 25 };
...
SingleItem[] order = { burger, fries, drink, wrap, shake };

foreach (SingleItem item in order)
{
    Console.WriteLine( item );
}
```

# Challenges

- ▶ How do we incorporate combos?
- ▶ Need a way to structure a recursive tree of elements and sub-elements
- ▶ Could we even support combo of combos?

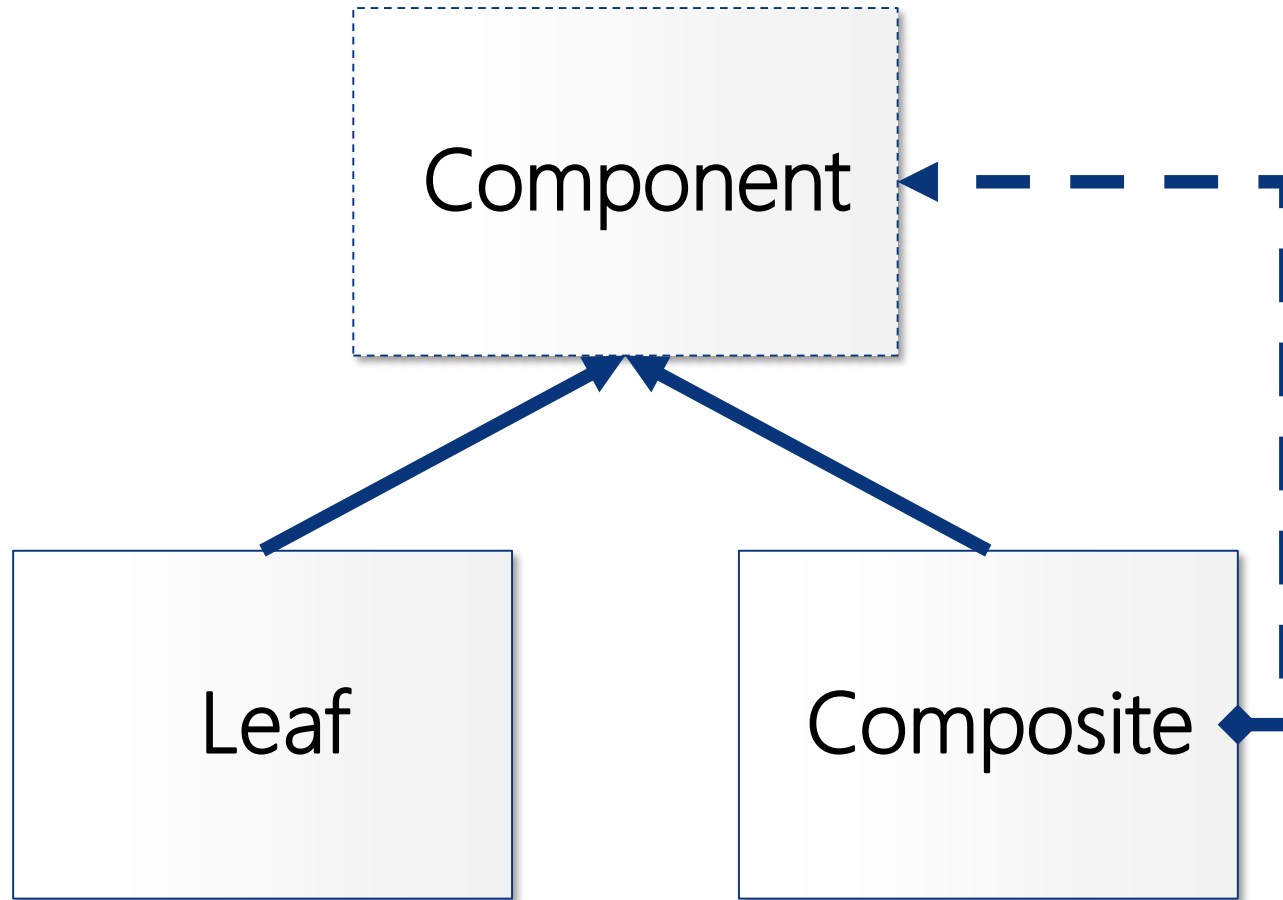


# Pattern: Composite

- ▶ *Compose objects into tree structures to present part/whole hierarchies. Composite lets clients treat individual objects and compositions of objects uniformly.*
- ▶ Outline
  - Define the elements of a recursive tree-like structure
  - Treats elements and groups of elements alike
- ▶ Origin: Gang of Four



# Overview of Composite Pattern



# Overview of Composite Pattern

- ▶ Component
  - Interface or abstract base class
  - Contains elements common to Leaf and Composite instances
- ▶ Leaf
  - Contains a “basic” element with no sub-elements
- ▶ Composite
  - Contains a “composite” element sub-elements



# Extensions

- ▶ The Composite Pattern can easily be generalized
  - Several distinct (or richer) Composite classes
  - Several distinct (or richer) Leaf classes
- ▶ Use the Iterator Pattern to “flatten” tree-like structure to sequences of Leaf instances







WINCUBATE

***Jesper Gulmann Henriksen***

PhD, MCT, MCSD, MCPD

Phone : +45 22 12 36 31

Email : [jgh@wincubate.net](mailto:jgh@wincubate.net)

WWW : <http://www.wincubate.net>

Hasselvangel 243

8355 Solbjerg

Denmark