Factory Method Pattern

Bryan Hansen
Director of Software Development

@bh5k

Concepts

Doesn’t expose instantiation logic

Defer to subclasses

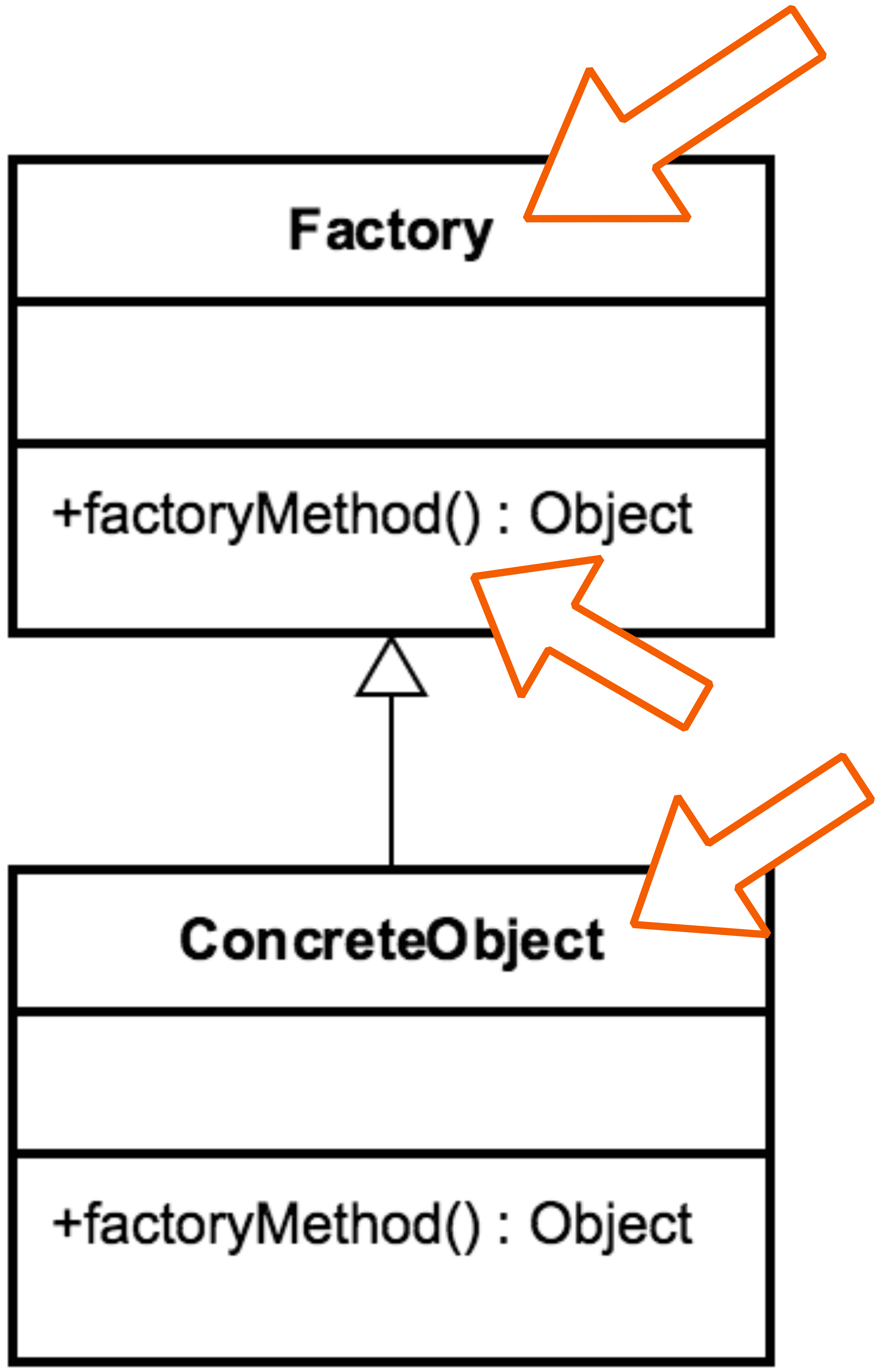
Common interface

Specified by architecture, implemented by user

Example:

- Calendar
- ResourceBundle
- NumberFormat

Design



Factory is responsible for lifecycle

Common Interface

Concrete Classes

Parameterized create method

Everyday Example - Calendar

```
Calendar cal = Calendar.getInstance();

System.out.println(cal);

System.out.println(cal.get(Calendar.DAY_OF_MONTH));
```

Demo

Create Pages

Create Website

Create Concrete Classes

Create Factory

Enum

Pitfalls

Complexity

Creation in subclass

Refactoring

Contrast

Singleton	Factory
Returns same instance	Returns various instances
One constructor method - no args	Multiple constructors
No Interface	Interface driven
No Subclasses	Subclasses
	Adaptable to environment more easily

Summary

Parameter Driven

Solves complex creation

A little complex

Opposite of a Singleton