Factory Method Pattern

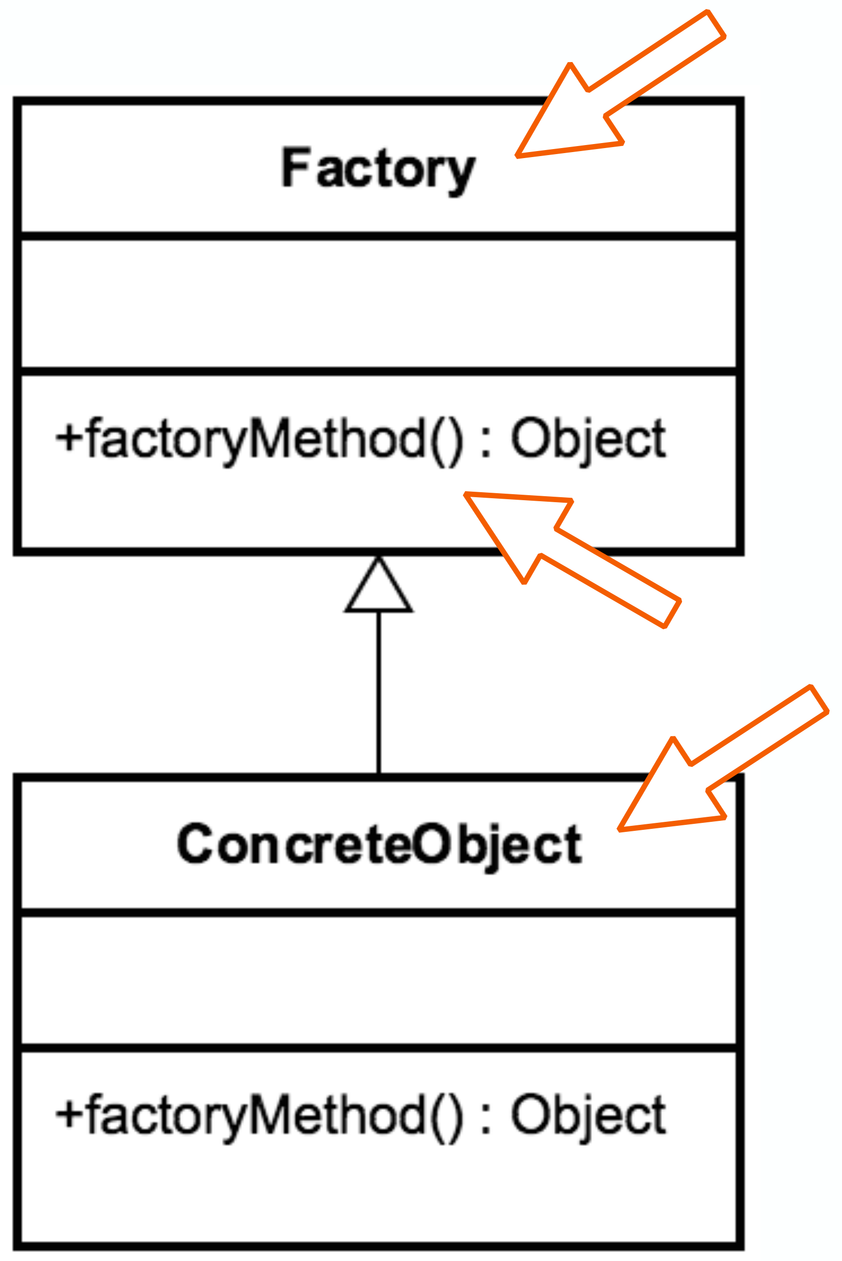
Bryan Hansen

twitter: bh5k | http://www.linkedin.com/in/hansenbryan

Concepts

- Doesn’t expose instantiation logic
- Defer to subclasses
- Common interface
- Specified by architecture, implemented by user
- Examples:
  - 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));
```

Exercise Factory

Create Pages

Create Website

Create Concrete Classes

Create Factory

Enum

Pitfalls

- Complexity
- Creation in subclass
- Refactoring

Contrast

Singleton

- Returns same instance
  - One constructor method - no args
- No Interface
- No Subclasses

Factory

- Returns various instances
  - Multiple constructors
- Interface driven
- Subclasses
- Adaptable to environment more easily

Factory Summary

- Parameter Driven
- Solves complex creation
- A little complex
- Opposite of a Singleton