The Factory Method Pattern

Structure & Functionality

Douglas C. Schmidt
Learning Objectives in This Lesson

- Recognize how the *Factory Method* pattern can be applied to extensibly create variabilities in the expression tree processing app.
- Understand the structure & functionality of the *Factory Method* pattern.
Structure & Functionality of the Factory Method Pattern

Douglas C. Schmidt
Factory Method

**Intent**

- Provide an API for creating an object, but leave the choice of the object’s concrete type to its derived class(es)

See [en.wikipedia.org/wiki/Factory_method_pattern](en.wikipedia.org/wiki/Factory_method_pattern)
Applicability

- When a class cannot anticipate the objects it must create.
**Applicability**

- When a class cannot anticipate the objects it must create.
- A class wants its derived classes to specify the objects it creates.
Applicability

- When a class cannot anticipate the objects it must create.
- A class wants its derived classes to specify the objects it creates.
- This approach is optional.
Applicability

- When a class cannot anticipate the objects it must create.
- A class wants its derived classes to specify the objects it creates.
- This approach is optional.
- An alternative is to pass a parameter to the factory method.
Applicability

- When a class cannot anticipate the objects it must create.
- A class wants its derived classes to specify the objects it creates.
- Or there’s a need to decouple the creation of an object from its subsequent use.

This decoupling is really the essence of the Factory Method pattern!
Structure & participants

- **User_Command**

  - **Product**
    - **ConcreteProduct**
    - **ConcreteCreator**
      - `FactoryMethod()`
      - `AnOperation()`
    - `FactoryMethod()`
      - `product = FactoryMethod()`
      - `return new ConcreteProduct`
Factory Method

GoF Class Creational

Structure & participants

- **Product**
- **ConcreteProduct**
- **Creator**
  - `FactoryMethod()`
  - `AnOperation()`
- **ConcreteCreator**
  - `FactoryMethod()`

```
... product = FactoryMethod()
...
 return new ConcreteProduct
```
Factory Method  GoF Class Creational

Structure & participants

Product

Creator
- FactoryMethod()
- AnOperation()

ConcreteCreator
- FactoryMethod()

ConcreteProduct

Eval_Command, Print_Command, Macro_Command, etc.

... product = FactoryMethod()
...

return new ConcreteProduct
Our app passes a string to the factory method rather than using a derived class.