**3. Factory Method**

**Tên, phân loại:**

* Tên chính thức: Factory Method
* Phân loại: Creational Pattern

**Mục đích, ý định:**

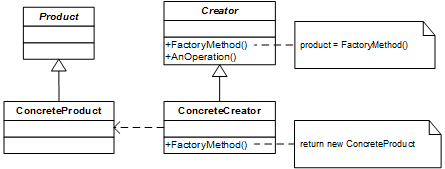
* Tạo ra một đối tượng từ một phương thức gọi là FactoryMethod trong một lớp trừu tượng. Đối tượng này có thể được kế thừa từ các lớp con và các lớp con có thể override lại hàm FactoryMethod để chỉ rõ đối tượng nào được tạo ra

**Khi nào sử dụng:**

* Khi một sản phẩm lại phân thành các sản phẩm con.
* Khi có nhu cầu sử dụng riêng cho các sản phẩm con.
* Khi một lớp sử dụng các con của nó để xác định đối tượng nào sẽ được tạo.
* Bạn muốn biết nội bộ hóa cách nhận biết làm thế nào biết đối tượng được khởi tạo thuộc về lớp nào.
* Khi có một vài biến thể của các mẫu factory cần được nhận diện.

**Khả năng ứng dụng:**

**Cấu trúc:**

****

**Các thành viên:**

* Các thành phần tham gia vào mẫu:
  + Product: định nghĩa một giao diện chung cho các đối tượng sẽ được Factory Method tạo ra.
  + ConcreteProduct: hiện thực hóa giao diện Product, là các sản phẩm sẽ được tạo ra.
  + Creator: một abstract class chung cho các ConcreteCreator, nó mô tả hàm trả về một đối tượng Product. Có thể gọi hàm đó là hàm tạo để tạo ra đối tượng Product.
  + ConcreteCreator override hàm tạo của Creator để tạo ra các đối tượng ConcreateProduct tương ứng.

**Sự cộng tác:**

* Product: đem lại một sự thống nhất chung cho các đối tượng được tạo ra từ các lớp kế thừa nó.
* Creator: là lớp cha của các creator con có nhiệm vụ tạo ra các product tương ứng.

**Các hệ quả mang lại:**

* Tách biệt định nghĩa của các lớp trong cùng cây phân cấp và ứng dụng sử dụng các thể hiện được tạp ra từ chúng.
* Khi cây phân cấp các lớp có sự thay đổi, sự ảnh hưởng đến các lớp sử dụng chúng bị giảm thiểu.
* Tạo ra một placeholder cho các đối tượng cùng cây phân cấp; bình thường nếu ta tạo đối tượng (thuộc lớp A) trực tiếp, sau đó nếu t muốn thay bằng một đối tượng khác (lớp của đối tượng này kế thừa từ lớp A) thì bình thường rất khó (vi phạm nguyên tắc Open-Closed). Khi dùng Factory ta dễ dàng thay thế đối tượng ban đầu vì nó không quan tâm đến kiểu cụ thể, chỉ cần sửa đổi factory tương ứng.
* Nếu muốn dùng Factory method và đem lại hiệu quả, bắt buộc lớp (có đối tượng cần tạo bằng Factory) cần được mở rộng từ một lớp cha chung.

**Chú ý liên quan đến việc cài đặt:**

* Lớp cơ sở là lớp abstract, và mẫu phải trả về một lớp làm việc hoàn chỉnh
* Lớp cơ sở chứa các phương thức mặc định, và các phương thức này được gọi nếu các các phương thức mặc định không đủ sử dụng.
* Các tham số được truyền vào factory nói cho nó biết kiểu đối tượng nào sẽ được nó trả về. Trong trường hợp này nhiều lớp sẽ cùng chia sẻ một phương thức cùng tên nhưng sẽ có các hành vi khác nhau.

**Ví dụ về các hệ thống thực tế:**

**Các mẫu liên quan:**

**Mã nguồn minh họa:**