Factory Method en PowerShell

En diseño de software, el patrón de diseño Factory Method consiste en utilizar una clase constructora (al estilo del Abstract Factory) abstracta con unos cuantos métodos definidos y otro(s) abstracto(s): el dedicado a la construcción de objetos de un subtipo de un tipo determinado. Es una simplificación en la que la clase abstracta tiene métodos concretos que usan algunos de los abstractos; según usemos una u otra hija de esta clase abstracta, tendremos uno u otro comportamiento.

Script de ejemplo de Factory Method en PowerShell:

[crayon-6a9d48007a716622655891/]

```
>> # Los m@@todos/propiedades est@@ticos demuestran los conceptos de almacenamiento/ recuperaci@@n.
>> #>
PS /Users/lamac> class FabricaBotes
>> {
>>     # Declarar y acceder a los colores nuevos
>>     static [BoteColor[]] $BoteColor
>>
>>     static [Object] getByType([Object] $O)
>>     {
>>         return [FabricaBotes]::BoteColor.Where({$_ -is $O})
>>     }
>>
>>     static [Object] getByName([String] $Nombre)
>>     {
>>         return [FabricaBotes]::BoteColor.Where({$_.Nombre -eq $Nombre})
>>     }
>>
>>     # Crear un m@@todo para la mezcla de colores: Mezclar
>>     [BoteColor] CrearBote([String] $Nombre, [String] $Type)
>>     {
>>         return (New-Object -TypeName "$Type" -ArgumentList $Nombre)
>>     }
>> }
PS /Users/lamac>
PS /Users/lamac> # Ejemplo de ejecuci@@n
PS /Users/lamac>
PS /Users/lamac> [FabricaBotes]$Fabrica = [FabricaBotes]::new()
PS /Users/lamac> [BoteColor]$Bote1 = $Fabrica.CrearBote("Amarillo","Amarillos")
PS /Users/lamac> [BoteColor]$Bote2 = $Fabrica.CrearBote("Azul","Azules")
PS /Users/lamac> $Bote1.Abrir()
Abierto el bote de pintura: Amarillo
PS /Users/lamac> $Bote2.Abrir()
Abierto el bote de pintura: Azul
PS /Users/lamac>
```