**FUNCIONAMIENTO DE EL PATRÓN DE CREACIÓN**

Los patrones creacionales tratan con las formas de crear instancias de objetos. El objetivo de estos patrones es de abstraer el proceso de instanciación y ocultar los detalles de cómo los objetos son creados o inicializados. Los creacionales están basados en dos conceptos:

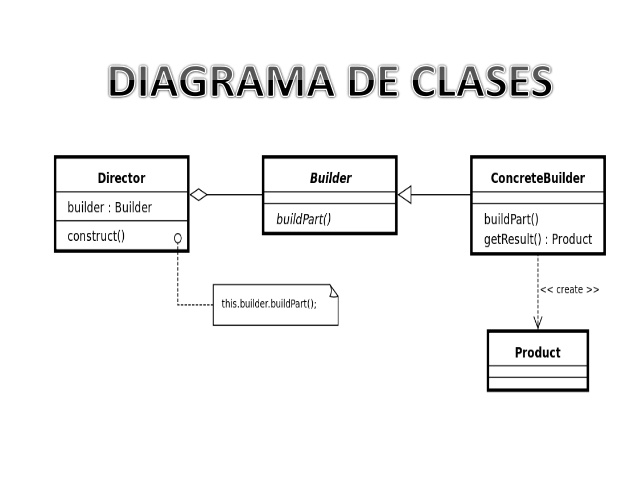
* Encapsular el conocimiento acerca de los tipos concretos que nuestro sistema utiliza. Se suelen trabajar con interfaces, por lo que la implementación que utilicemos quedará aislada.
* Ocultar como estas implementaciones concretas necesitan ser creadas y cómo se combinan entre sí.

También, este tipo de patrón tiene varias subtipos:

* Abstract Factory: permite trabajar con objetos de diferentes familias de manera que no se mezclen entre sí, para conseguir que el tipo de familia que se esté utilizando sea transparente.
* Builder: abstrae el proceso de creación de los objetos complejos, centralizando en un solo punto.
* Factory Method: centraliza en un constructor la creación de objetos de un tipo determinado.
* Prototype: crea un objeto a partir de la clonación de un objeto ya existente.
* Singleton: garantiza que solo exista una instancia de un objeto y que la forma de acceder a dicha instancia sea general

Es interesante utilizar patrones porque te ayudan a ahorrar tiempo, estar seguro de que el código es correcto, lo cual en programación es una gran ventaja, establecen un lenguaje común. Por ejemplo ¿Cuando podemos aplicar el patrón Abstract Factory?. Cuando un sistema debe ser independiente de cómo sus objetos son creados, debe ser configurado con una cierta familia de productos y cuando se necesita reforzar la noción de la dependencia mutua entre ciertos objetos.

Ejemplo patrón creacional Builder:



Para el ejercicio de la práctica, he imaginado como si fuese una Pizzería, con las clases indicadas en la siguiente foto

* Cocina: Director
* HawaiPizzaBuilder: Concrete Builder
* PicantePizzaBuilder: Concrete Builder
* Pizza: Clase del producto pizzas
* Pizza Builder: ConcreteBuilder
* Principal: Main

