Resuemen de Frameworks

Definicion completa de framework:

Un framework se puede ver como el resultado de refactorizar una aplicación, no solo para mejorar su calidad interna sino para abstraer lo que es común en todas las familias de aplicaciones similares. En ese proceso de refactoring se identifica lo que se mantendrá constante y será, a partir de ese momento, responsabilidad de los desarrolladores del framework y lo que podrá variar de aplicación a aplicación y será responsabilidad de los desarrolladores de aplicaciones que usen el framework.

 Para separar la implementación de la parte constante  (a lo que llamamos **frozenspot)** y de las partes que varían (a las que llamamos **hotspots)**, se introducen en el framework puntos de extensión. Estos tienen la forma de plantillas que ofrecen ganchos para que, mediante herencia o composición, los programadores de aplicaciones definan el comportamiento variable. Algunos patrones de diseño (particularmente el template method y el strategy) son frecuentemente utilizados para introducir esos puntos de extensión.

 La construcción y uso de frameworks es una estrategia para modularizar y reutilizar, que mejora la calidad del software y aumenta la productividad de quienes lo hacen. Un framework, visto como módulo de software, no solo encapsula operaciones y estructuras de datos reusables sino que las combina para resolver, con demostrada eficacia, los requerimientos comunes a una familia de aplicaciones. En pocas palabras, encapsula el diseño y la implementación de la parte más compleja de dichas aplicaciones y permite su reutilización.

Un framework es en sí mismo un módulo de software reutilizable. Muchas aplicaciones se construirán sobre él. El framework será independiente de las aplicaciones. Sin embargo, las aplicaciones dependerán de que la interface (conjunto de puntos de extensión) y comportamiento del framework se mantenga estable. Esto es importante dado que, a lo largo de su vida, el framework será foco de contínuo refactoring y de mejoras para eliminar bugs y hacerlo más efectivo.

Servidores TCP:

Es un programa que implementa una de las partes de la arquitectura cliente/servidor. Un servidor escucha un socket TCP, un cliente establece una conexión con el servidor para enviar mensajes (plain text) y el servidor responde. Ej: SMTPServer (el que usamos en la pagina para enviar mails).

Flujo de control:

1.Crear un socket (IP\_address +port)

2.Escucha un socket

3.Aceptar al cliente => sesion

4.Recibir un mensaje

a.condicion => responder (según el server, la respuesta sera distinta).

b.!condicion

b1. Cerrar sesion

b2. Goto 2 (?)

Este “flujo de control” se va a reusar pero es incompleto (generalmente no compila), entiendo que es flujo debe poder “responder” en donde se puede “enganchar” la funcionalidad. El flujo pertenece al framework.

Hook: Es un mecanismo que permite **"enganchar"** o **insertar código personalizado** en puntos específicos del ciclo de vida de una aplicación o componente, **sin modificar el código base del framework**.

Tambien se puede definir como un punto de entrada proporcionado por un framework que permite al desarrollador **inyectar lógica personalizada** sin modificar el núcleo del framework, facilitando la extensibilidad y modularidad del software.

Framework de caja blanca:

Definicion: Un **framework de caja blanca** es un marco de trabajo en el que el control del flujo de ejecución es proporcionado por el framework, pero **el desarrollador puede intervenir y modificar su comportamiento interno extendiendo clases base o sobrescribiendo métodos.**

.La instanciacion hereda y completa el loop de control, es posible que requiera agregar metodos a clases del framework y demanda conocimiento del codigo del framework.

.El desarrollador conoce la estructura interna del framework (de ahí “caja blanca”).

.Se basa en el principio de inversión de control: el framework llama al código del desarrollador, no al revés.

Framework de caja negra:

Definicion: Un **framework de caja negra** es aquel en el que el desarrollador **utiliza y configura sus componentes sin necesidad de conocer o modificar su funcionamiento interno**. El framework ofrece una **interfaz pública clara y estable**, y el desarrollador se limita a **componer y configurar** esos componentes.

.El desarrollador no necesita (ni puede) intervenir directamente en la lógica interna del framework.

.Se basa en el uso de composición y configuración, en lugar de herencia.

.En general, es más fácil de usar y menos propenso a errores, pero menos flexible que uno de caja blanca.

.Suele seguir el principio de "convención sobre configuración".

**FrozenSpot**: aspecto del framework que afecta a todas las instanciaciones y que no se puede modificar

**HotSpot**: estructura en el codigo que permite modificar el comportamiento del framework, para instanciar y para extender.