



Frameworks

Composición, herencia y hotspots

Hotspots y Frozenpots

- Framework hotspots: partes del artefacto a construir con el framework, que puedo cambiar
 - Ej: como se formatearán los mensajes de log en `java.util.logging`
 - Ej: preparación del fixture en tests con JUnit (`@BeforeEach`)
- Framework frozenspot: partes que seran igual para todos los artefactos construídos con el framework
 - Ej. Como se obtienen y organizan los Loggers en `java.util.logging`
 - Ej. Cómo/dónde/cuándo indicar que se espera como resultado de cada operación con JUnit



Instanciando hotspots

- Un framework frecuentemente ofrece muchos hotpost
- Cuando aprovechamos/usamos un hotspot decimos que “instanciamos un hostpot”
- Algunos “debo” instanciar
 - p.e. Loggers en `java.util.logging`
- Otros “puedo” instanciar
 - p.e., Formaters y handlers en `java.util.logging`



¿Pero dónde está el hotspot?



- El concepto de Hotspot refiere, de manera abstracta, a un aspecto de puedo cambiar
- Reconocerlos nos permite diseñar mejor el framework y las aplicaciones que lo usan^(*)
- Instanciar un hotspot puede requerir una combinación acciones, de herencia y de composición
- Hacer algo interesante puede requerir aprovechar varios hotspots
- La decisión de cómo se instancia un hotspot implica cosas diferentes para el desarrollador del framework y para quienes lo usan

(*) W. Pree, "Hot-spot-driven framework development," in Summer School on Reusable Architectures in Object-Oriented software Development, 1995, pp. 123–127.

Como usuarios del framework



- Herencia

- Implemento, extiendo, y redefino métodos
- Uso variables y métodos heredados
- Podría cambiar cosas que el desarrollador no tuvo en cuenta
- Puedo extender el framework
- Debo aprender qué heredo y qué puedo hacer con ello
- No puedo heredar comportamiento de otro lado

- Composición

- Instancio y configuro
- Conecto a mi código con callbacks
- Solo conozco algunas clases del framework y sus mensajes
- Mis objetos son mis objetos, heredo de donde quiero
- No puedo cambiar o extender el framework
- Solo tengo acceso a los objetos que recibo en los callbacks

Como desarrolladores del framework



- Herencia
 - Dejo ganchos en clases del framework (abstractas y concretas)
 - Paso el control con mensajes a self
 - Se que esperar del código del usuario (si hace lo que digo)
 - No necesito pensar todos los hotspots y todos sus casos
 - No necesito pasar estado como parámetros (está en las v.i.)
 - Debo documentar claramente que se puede tocar y que no
 - No puedo cambiar el diseño sin preocuparme por los usuarios
- Composición
 - Dejo objetos configurables para ajustar el comportamiento
 - Paso el control con callbacks
 - Debo pensar todos los hotspots y sus casos
 - Debo pasar/actualizar todo el estado necesario en los callbacks
 - No se nada del código del usuario
 - No necesito explicar (mucho) como funciona
 - Puedo cambiar mi diseño sin preocuparme (mucho) por los usuarios

Herencia / composición / blanco / negro



- Un framework tendrá hotspots que se instancian con herencia y otros que se instancian por composición
- Por lo general, arrancan dependiendo mucho de herencia (caja blanca) para ir evolucionando a composición (caja negra)
- Es más fácil desarrollarlos si son caja blanca, y usarlos si son caja negra
- Es más desafiante desarrollarlos si son caja negra, y usarlos si son caja blanca

