Patrones de diseño

Tópicos Selectos de Computación I
9 de Agosto de 2013

Agenda

- **Introducción**
- Origen
- Objetivo y definición
- Elementos
- Clasificación

Introducción (1)

- El aprendizaje del desarrollo de software es difícil.
- Lenguaje y frameworks son complejos.
- Medio de comunicación mejor.
- Patrones de diseño y los lenguajes asociados pueden ser una buena propuesta.

Introducción (2)

- ¿Qué es un patrón?
  - Una solución a un problema en un contexto.
  - Una manera estructurada de representar información acerca del diseño.
  - Una manera de comunicar el diseño de los expertos a los novatos.
  - Una clave para realmente entender OO.

Introducción (3)

- ¿Qué es un patrón?
  - Proveen soluciones a problemas OO comunes.
  - Transfieren experiencia a los novatos.
  - Vocabulario común para los diseñadores.
  - Documentación para frameworks existentes.
  - Con frecuencia permiten crear diseños más sofisticados y seguros.

Agenda

- Introducción
- **Origen**
- Objetivo y definición
- Elementos
- Clasificación
- Ejemplo

Origen

- Los patrones provienen de la arquitectura.
- Un lenguaje de patrones describe:
  - Motivos comunes de arquitectura.
  - Cómo lograr abstraerlos para poder llevarlos a varios ambientes o contextos.

Agenda

- Introducción
- Origen
- **Objetivo y definición**
- Elementos
- Clasificación

Definiciones (1)

- Cada patrón es una regla de 3 partes, que expresa una relación de un contexto, un sistema de fuerzas que ocurren repetidamente en ese contexto y una configuración de software que permite que se resuelvan esas fuerzas. A Timeless Way of Hacking".

Definiciones (2)

- Una descripción de un problema recurrente que ocurre en un contexto determinado y, basado en un conjunto de fuerzas, recomienda una solución. La solución es usualmente un mecanismo simple: una colaboración entre dos o más clases, objetos, servicios, procesos, threads, componentes o nodos que trabajan juntos para resolver el problema identificado por el patrón. *Enterprise Development Reference Architecture.*

Definiciones (3)

- Los patrones ayudan a construir sobre la experiencia colectiva de ingenieros de software experimentados. Estos capturan la experiencia existente y que ha demostrado ser exitosa en el desarrollo de software, además ayudan a promover las buenas prácticas del diseño. Cada patrón aborda un problema específico y recurrente en el diseño o implementación de un sistema de software. *Pattern Oriented Software Architectura, Volume 1*.

Objetivo y definición

- Objetivo:
  - Reutilización del conocimiento:
    - Cada patrón describe un problema que ocurre cada una y otra vez en nuestro entorno y describe también el núcleo de la solución al problema, de forma que puede utilizarse un millón de veces sin tener que hacer dos veces lo mismo.
    - Definición:
      - Un patrón es una solución a un problema general que puede adaptarse a un problema concreto.

Agenda

- Introducción
- Origen
- Objetivo y definición
- **Elementos**
- Clasificación
- Ejemplo

Patrones de diseño – Elementos

- **Nombre del patrón**: el nombre del patrón y la referencia dónde el fue inicialmente propuesto.
- **Sinopsis**: Una breve descripción del patrón.
- **Contexto**: Una descripción del problema que pretende resolver.
- **Causas**: Lista las razones y motivos que afectan el problema y la solución. La lista de causas ilumina las razones por las que uno podría haber elegido utilizar el patrón y proporciona una justificación de su uso.

Patrones de diseño – Elementos

- **Solución**: Describe brevemente la solución y sus elementos en más detalle. La selección de la solución contiene dos sub-secciones:
  - **Estructura**: Utiliza diagramas de clases UML para mostrar la estructura básica de la solución. Los diagramas de secuencias UML de esta sección presentan mecanismos dinámicos de la solución. Hay una explicación detallada de los participantes y los colaboradores.
  - **Estrategias**: Describe diferentes formas en las que se podría implementar el patrón.

Patrones de diseño – Elementos

- **Consecuencias**: Implicaciones de uso del patrón.
- **Implementación**: Detalles de implementación a considerar.
  - **Java API**: Cuando se dispone, un ejemplo del API es conveniente.
  - **Ejemplo del código**: Un código de ejemplo en Java.
- **Patrones relacionados**: Una lista de patrones relacionados.

Agenda

- Introducción
- Origen
- Objetivo y definición
- Elementos
- **Clasificación**

Clasificación

- **Patrones de creación**: relativo a la creación de objetos.
- **Patrones estructurales**: trata de la composición de clases u objetos.
- **Patrones de comportamiento**: caracteriza la forma en que clases y objetos interactúan y la distribución de las responsabilidades.

Clasificación

- **Patrones de creación:**
  - Abstract factory, builder, factory method, prototype, singleton.
- **Patrones estructurales**
  - Adapter, bridge, composite, decorator, facade, flyweight, proxy.
- **Patrones de comportamiento**
  - Chain of responsibility, command, interpreter, iterator, mediator, memento, observer, state, strategy, template method, visitor.