Patrones de Diseño: Patrones de Comportamiento. Tema 5-4: Interpreter

Descripción del patrón

Nombre:

- Intérprete
- Propiedades:
 - Tipo: comportamiento
 - Nivel: clase
- Objetivo o Propósito:
 - Definir un intérprete para un lenguaje sencillo.





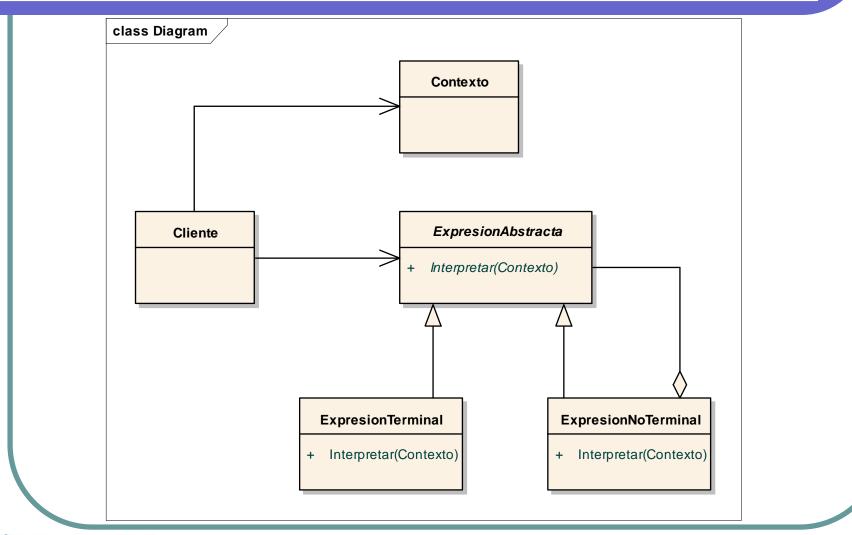
Aplicabilidad

- Use el patrón Interpreter cuando:
 - Haya que interpretar un lenguaje sencillo. Para gramáticas complejas, la jerarquía de clases para la gramática se convierte en algo grande e inmanejable.
 - Podemos representar las sentencias del lenguaje como árboles abstractos de sintaxis (AST) -Abstract Syntax Trees.
 - La resolución del problema a evaluar pueda ser expresada en términos de ese lenguaje.
 - La eficiencia no sea un aspecto fundamental.





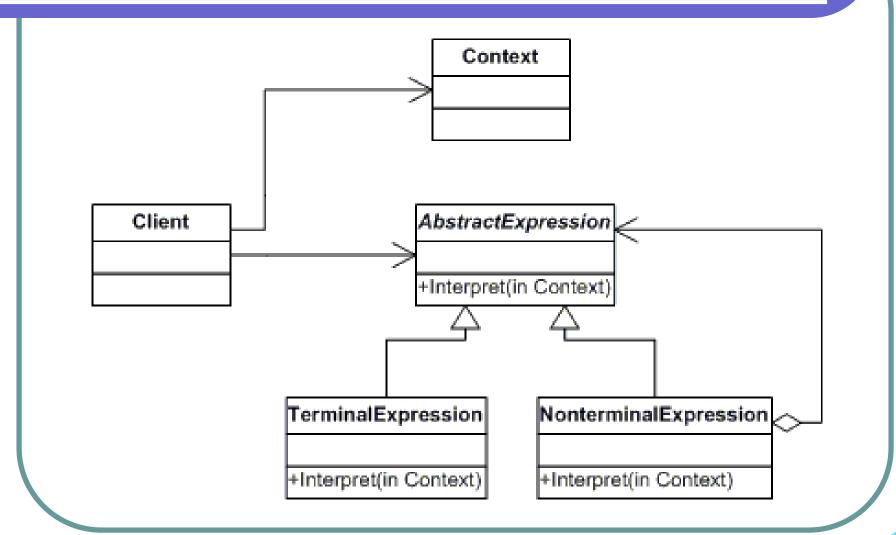
Estructura







Estructura







Estructura. Participantes

- ExpresionAbstracta: Clase abstracta o interfaz a través de la cual el cliente interactúa con las expresiones.
- **ExpresionTerminal:** Implementación del método *interpretar* para los nodos terminales en la gramática y en el árbol de sintaxis.
- **EspresionNoTerminal:** Implementación del método *interpretar* para los nodos no terminales. Mantiene una referencia a la siguiente expresión e invoca el método *interpretar* en cada uno de sus hijos.
- Contexto: Contiene información global que se necesita en distintos lugares del interprete. Es un canal de comunicación entre distintas instancias de ExpresionAbstracta.
- Cliente: Construye o recibe una instancia de un árbol de sintaxis abstracta. Este árbol de sintaxis se compone de instancias de expresiones terminales y expresiones no terminales que permiten modelar una sentencia específica. Invoca al método interpretar.





Consecuencias

- La gran ventaja del patrón Intérprete es que la gramática puede ser ampliada o cambiada muy fácilmente. Para añadir una regla, solo hay que crear otra clase que implemente la interfaz ExpresionAbstracta. Esta clase implementará una nueva regla en el método *interpretar*.
- El patrón Intérprete, en cambio, no es adecuado cuando la gramática es grande ya que puede generar problemas para depurar y mantener el sistema.





Patrones relacionados

- Composite: La estructura de las expresiones interpretadas se basa en este patrón con expresiones terminales (nodos hoja) y no terminales (nodos rama).
- Flyweight: Se puede aplicar en algunas de las expresiones, para reducir el número de objetos similares.
- Iterator: Usado para recorrer el árbol.
- **Visitor:** Para mantener el comportamiento de cada nodo del árbol en una clase.



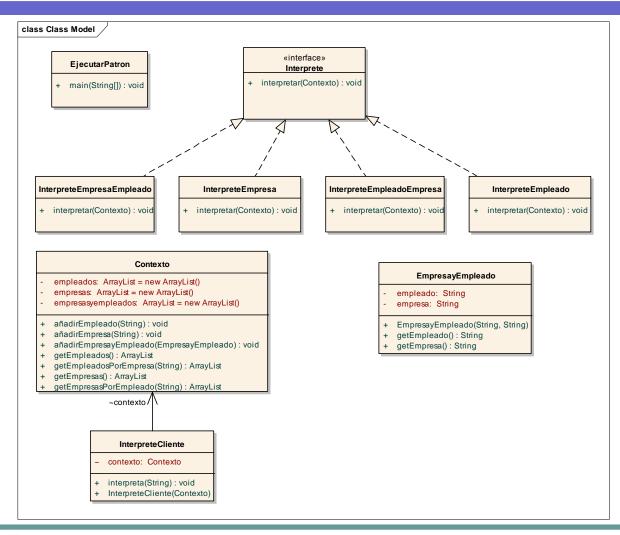


Código de ejemplo

Empresas y Empleados



Código de ejemplo







Código de ejemplo

- Identificamos a continuación los elementos del patrón:
 - ExpresionAbstracta: Interprete.
 - ExpresionTerminal: InterpreteEmpresaEmpleado, InterpreteEmpresa, InterpreteEmpleadoEmpresa, InterpreteEmpleado.
 - Contexto: Contexto.
 - Cliente: InterpreteCliente.



