Design patronen in C#: Inleiding

Binnen programmeren kom je allerlei scenario’s tegen welke je via allerlei wegen kunt oplossen. Design patronen zijn herbruikbare oplossingen om deze scenario’s en issues op te lossen. Welk design patroon je gebruikt, is afhankelijk van het scenario en het probleem waar een programmeur tegenaan loopt.

Het gebruik van design patronen is als eerste beschreven door Christopher Alexander in zijn boek ‘A Pattern Language: Towns, Buildings, Construction’. In het boek beschrijft hij hoe je een “taal” kunt gebruiken voor het ontwerpen van een stad. Deze “taal” bestaat uit verschillende patronen. De patronen beschrijven bijvoorbeeld hoe hoog de deuren zouden moeten zijn, hoe hoog een gebouw moet zijn, etc. Dit idee is verder opgepakt door vier auteurs, Erich Gamma, John Vlissides, Ralph Johnson en Richard Helm. In 1994, publiceerden zij het boek “Design Patterns: Elements of Reusable Object-Oriented Software”.

In dit boek worden in totaal 23 design patronen beschreven, ook wel bekend als de ‘Gang of Four’ design patronen. Deze patronen worden ingedeeld op basis van het doel van het patroon. De verschillende design patronen zijn op te delen in 3 hoofdcategorieën:

1. **Creatie ontwerppatroon**: Dit ontwerppatroon biedt mechanismen voor het maken van objecten welke de flexibiliteit en herbruikbaarheid van de code bevorderen.
2. **Structureel ontwerppatroon**: deze patronen leggen vast hoe objecten en klassen in grotere structuren kunnen worden geassembleerd, terwijl de structuren flexibel en efficiënt blijven.
3. **Gedrag ontwerppatroon**: Dit patroon zorgt voor effectieve communicatie en toewijzing van verantwoordelijkheden tussen objecten.

Welk patroon je gebruikt, hangt geheel af van het mogelijke probleem waar je tegen aanloopt en welke je wilt oplossen. Let wel, het gebruik van patronen moet het programmeren nu en later onderhouden van de code makkelijker maken. Een patroon kan de code namelijk ook complexer maken dan nodig is waardoor het juist het tegenovergestelde effect heeft. Denk dus altijd goed na of er een design patroon is welke bij het issue waar je tegenaan loopt past en of het probleem hierdoor makkelijker wordt. Zo niet, gebruik dan geen patroon.