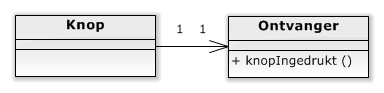
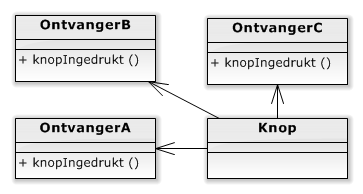
## Het Listener-Pattern

Het Listener-Pattern is een patroon dat gebruikt kan worden om een leverancier van signalen met een of meerdere ontvangers te verbinden, zonder dat de leverancier alle ontvangers moet kennen.

Als voorbeeld beginnen we met een veel voorkomende situatie: Een knop moet aan een ander object door kunnen geven dat hij is ingedrukt. In dit geval is het mogelijk om simpelweg de knop het ontvangend object te laten kennen, en een functie aan te roepen wanneer de knop is ingedrukt.

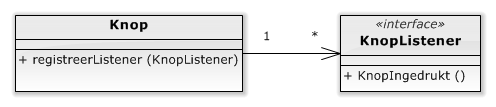


Echter, als er in de toekomst meerdere ontvangers worden toegevoegd, moet ook de knop-klasse worden aangepast, waardoor het ook moeilijker wordt om in een ander project de zelfde knop-klasse te gebruiken, zonder direct ook alle ontvangers mee te moeten nemen.

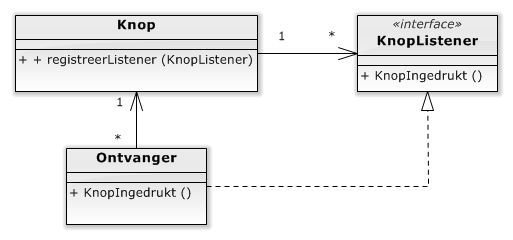


Om het zowel in de toekomst makkelijker te maken om nieuwe ontvangers toe te voegen, danwel de code voor de knop te kunnen hergebruiken, kan het Listener-pattern toegepast worden.

Bij het Listener-pattern wordt er gebruik gemaakt van een Interface die een functie definieert die door de zender (in het voorbeeld de Knop,) wordt aangeroepen voor alle geregistreerde ontvangers. Hierdoor kan de relatie tussen zender en ontvanger worden omgedraaid; de zenders kennen de ontvanger, maar de ontvanger kent de zenders alleen via de interface.

Voor de implementatie van het Listener-pattern in het voorbeeld, moet allereerst een interface gedefinieerd worden, die bekend is bij de Knop en waarin een functie is vastgelegd die de Knop kan aanroepen als er een event plaatsvind, bijvoorbeeld wanneer de Knop wordt ingedrukt. Daarnaast moet er een manier zijn voor nieuwe listeners om zich bij de Knop te registreren.

In deze situatie hoeft een Ontvanger alleen maar de interface KnopListener implementeren, zodat de Knop objecten van deze klasse kan accepteren via registreerListener.



Zoals in het schema te zien is, hoeft de Knop de Ontvanger niet meer te kennen, maar kan de ontvanger nog steeds events van de Knop ontvangen. Ook kan de Knop nu events sturen naar andere klassen zonder direct deze klassen te kennen, waardoor het makkelijker is om de code te hergebruiken in andere projecten.