Stijn Heynderickx K. Soontjes

Piotr Mazurek 26-5-2016

MHP Project Digital Broadcasting

Voor ons project hebben wij gekozen voor het spelletje “Brickball”, met als design pattern het Observer Pattern. Dit is een object-georiënteerd pattern dat ook in MVC-modellen wordt toegepast. Dit pattern wordt ook gebruikt bij een 1 op veel relatie tussen de objecten en is daarom perfect in ons spel omdat we steeds een update van de tijd moeten hebben van wat er op het scherm gebeurt. Hier zullen de objecten geobserveerd worden en wanneer een van de objecten verandert zullen alle andere objecten die ervan afhankelijk zijn mee geüpdatet worden.

Observer pattern gebruikt 3 hoofdklasses: ObserverInterface , Observer en Subject.

- ObserverInterface : dit is een abstracte klasse waar de basis update methoden bevat.

-Observer: implementeert(erft over van) ObserverInterface zodat die zijn update methode kan gebruiken. De observer kijkt of een object van staat verandert en voert een update-methode die een update van het object zelf uitvoert uit en die de andere observers laat weten dat het object van staat is verandert. De andere observers zullen de objecten die ze observeren dan ook updaten.

-SubjectInterface: dit is een abstracte klasse die de basismethoden bevat voor de subject.

-Subject: deze klasse implementeert(erft over van) de SubjectInterface zodat die de methoden kan gebruiken. Deze klasse voegt een observer toe of verwijdert die van objecten(register/unregister).

In onze main klasse (hellotvxlet) gebruiken we de subject klasse als publisher om de bal, de blokjes en de balk te een observer toe te voegen. Wanneer het “balletje” een blokje raakt zal het blokje geüpdatet worden en zal die van de scene worden afgehaald. Het “balletje” zal dan ook van richting veranderen. Al deze updates worden uitgevoerd door de observers. In de main klasse staan alle gamelogica zoals collision detection en het aanmaken en opzetten van alle blokken op de scene.