Kennismaking met Swing: event handlers.

In de gameStart methode van de AnimatiePanel Klasse wordt een aparte **thread** opgestart waarbinnen in een oneindige lus voortdurend een game update gebeurt (de beweeg methode van de klasse BewegendeCirkel wordt voor elke cirkel in het ballenveld aangeroepen) en de cirkels worden hertekend (de repaint methode van de JPanelklasse roept de paintComponent methode aan). De thread wordt telkens 4 milliseconden onderbroken (sleep(4))

De methode addNotify in de animatiePanel klasse zorgt ervoor dat bij het tekenen van de frame met zijn panels het animatiepanel de focus krijgt (requestFocus) zodat je bijvoorbeeld het panel kan laten reageren op een ingetypte toets op het toetsenbord

Eventhandlers:

In het animatiepanel willen we door te klikken met de linker muisknop ballen kunnen toevoegen. Via de methode **addMouseListener** geven we aan dat we in dit panel een actie willen koppelen telkens wanneer er met de muis in het panel geklikt wordt. Als parameter van de addMouseListener methode geven we een MouseAdapter object mee. Binnen de codeblok van MouseAdapter geven we aan op welke muisactie we willen reageren (klikken met de muis, slepen met de muis, …). In ons geval willen we kunnen reageren op het klikken van de muis binnen de pannel (**MouseClicked**). Deze methode krijgt als parameter een **MouseEvent** mee (gegenereerd door java als we klikken met de muis – dit mouseEvent bevat onder andere de x en y coördinaat waar geklikt is, …). Binnen de mouseClicked methode kunnen we info over het MouseEvent opvragen met behulp van getters (getX, …).

Willen we voorbeeld binnen ons panel kunnen reageren op het intypen van een toets, dan kunnen we een KeyListener toevoegen aan het panel en kunnen we reageren op KeyEvents.

Wanneer we willen actie koppelen aan het klikken op een JButton (zoals in het KnoppenPanel klikken op de Herstart Animatie knop) dan koppelen we aan die knop een ActionListener(**addActionListener**) en werken deze ActionListener als een inner class uit -> we schrijven de code voor te reageren op de knopklik uit in de **actionPerformed** methode die een **ActionEvent** als parameter heeft (dit ActionEvent object wordt weer door Java zelf aangemaakt telkens we op de knop klikken en bevat onder ander info over de tijd wanneer op knop isgeklikt, de naam van het aangeklikte knop object, …

Op een gelijkaardige manier kan je code koppelen aan een aangeklikte keuze in een JComboBox of een JSpinner of een ander Swing component.

Zoek in de Java documentatie telkens op welke listeners je kan koppelen aan het swingobject waar je wil reageren op een bepaalde actie, welke methodes je dan kan uitwerken voor die listener en welke events java genereert die je in je eventhandlers kan ondervragen.