Game Programming - opdrachten EP3

Voor deze opdrachten zijn dezelfde regels van toepassing als deze van EP2.Ook de workshopnota's en slides sluiten nog steeds aan bij onderstaande opdrachten, je neemt deze dus best eerst door. Je hoeft de opdrachten waarvoor je reeds een voldoende scoorde in EP2 niet opnieuw te maken. Je kan je behaalde punten zien bij het onderdeel Cijfers op Toledo. Voor elke opdracht verwachten we ook nu weer een verslag met bibliografie, zoals in de intro slides van EP2 beschreven staat.

De deadline voor deze opdrachten is **vrijdag 18 augustus 20:00u.**
Je maakt opnieuw gebruik van git voor het inzenden van je project. Je maakt een nieuw git project aan met de naam **1617GPEP3NaamVoornaam** en je kent aan de docenten (Katja Verbeeck, Tim Vermeulen en Joris Maervoet) masterrechten toe.
Voor deelname aan de mondelinge verdediging zal je jezelf moeten inschrijven (d.i. de uurregeling; de dag van het examen vind je op je examenrooster).
Deze Tolinto-lijsten worden half augustus opengesteld.

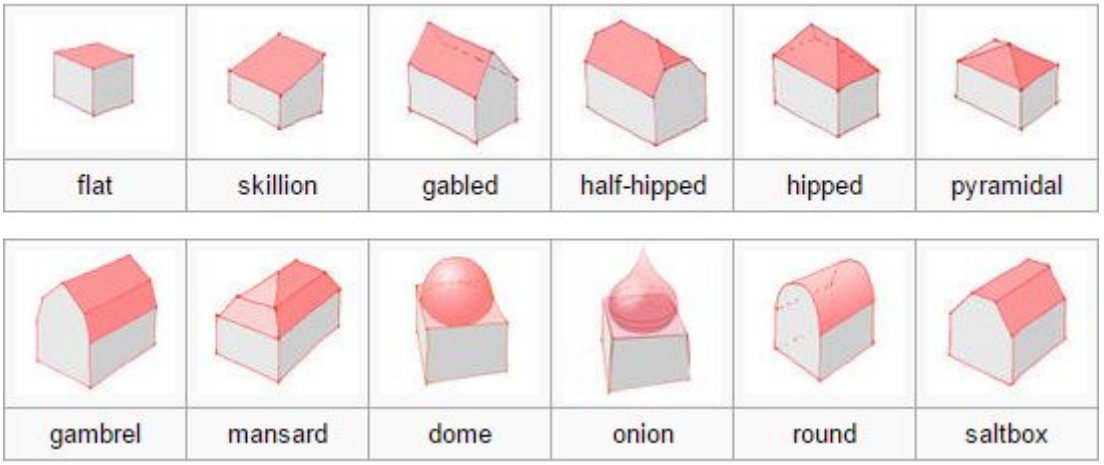
Veel succes !

Opdracht 1: 3D-huizenconfigurator

Maak in WPF een interactieve 3D-huizenconfigurator zoals hieronder beschreven staat. Voor de camera bediening mag je gebruik maken van een bibliotheek. Ook de binnenkant van het huis moet te bezichtigen zijn. Minstens volgende parameters kun je configureren:

- Hoogte, breedte en lengte van het huis (in cm)
- Het daktype (zie Figuur 1)
- Extra parameters voor de dakverhoudingen afhankelijk van het geselecteerde daktype

Ondersteun minstens 4 van de in Figuur 1 opgelijste daktypes, waarvan minstens 1 met gebogen vormen (dome, onion of round). Maak ook gebruik van textures.



Figuur 1: daktypes. Bron: http://wiki.openstreetmap.org/wiki/Simple_3D_buildings.

Deze specificaties kunnen als inspiratiebron dienen bij de verdere uitwerking van deze opdracht. Uiteraard is het helemaal niet de bedoeling om een interpreter te schrijven voor de building-specs van OpenStreetMap.

Opdracht 2: kortste pad voor robots met obstakels

In deze opdracht simuleer je a.d.h.v. GDI+ in C# een cirkelvormige robot die zich voortbeweegt in een wereld waarin zich enkele **rechthoekige obstakels** bevinden.

Deze opdracht bestaat uit 2 delen:

- **Padplanning (A.I.)**
  - Plan een zo kort mogelijk pad dat de robot kan afleggen tussen 2 opgegeven punten (= van A naar B).
  - Opgelegd algoritme: een RRT-variant naar keuze
  - De planning houdt rekening met de obstakels en de afmetingen van de robot.
- **Fysische simulatie**
  - Je simuleert de robot die zich voortbeweegt volgens de wetten van de fysica, bij voorkeur langs het geplande pad.
  - **Je voorziet een aan/uit bediening van de motor van de robot.** Deze genereert uitsluitend een constante kracht of geen kracht. Opgelet kracht is niet hetzelfde als snelheid!
  - De robot kan enkel vooruit rijden. De hoek waarin de robot rijdt kan ook aangepast worden met de pijltjes en wordt grafisch weergegeven in de simulatie.
  - Wanneer de robot niet goed bediend wordt kan het natuurlijk zijn dat deze toch botst op een obstakel of de grenzen van de robotwereld. Simuleer dan ook deze fysische botsing.
  - Je houdt ook rekening met de wrijvingskracht tussen de robot en het oppervlak waar hij zich op voortbeweegt. Hierdoor zal de robot vertragen en tot stilstand komen als geen kracht wordt gegeven door de motor. Maak mogelijk dat de wrijvingscoëfficiënt instelbaar is.

De wereld waarin je robot rondwandelt moet kunnen variëren. Maak hiervoor een configuratiefile waarin je de positie en afmetingen van de obstakels kan definiëren. Deze wordt dan ingeladen bij de start van je applicatie. Tijdens de demonstratie kunnen we dan eenvoudig een nieuwe wereld testen door deze configuratiefile aan te passen. Test dus zelf vooraf ook al verschillende configuratiefiles.