Prosjektbeskrivelse

Jeg valgte å jobbe med python sitt standard-bibliotek «socket» fordi jeg syntes nettverk mellom scripts virket som en interessant utfordring. Jeg har prøvd å jobbe med det før, men fikk det aldri helt til å fungere.

Oppstartsfasen

Under oppstarten så kodet jeg en veldig enkel server og enkelt spill. Disse filene brukte jeg til å test hvordan de ulike komponentene av socket.py fungerte. Jeg leste dokumentasjon for å være sikker på hva alle funksjonene gjorde. Jeg prøvde også å bli helt sikker på om jeg i det hele tatt ville klare å bli ferdig med en slik oppgave i tide.

Det gikk bra så derfor så valgte jeg å gjøre årsoppgaven til et enkelt spill med server og database kobling som kunne ha flere klienter samtidig. Det er det som har vært oppgaven, jeg har også prøvd å gjøre hele prosjektet objektorientert ved å skrive mine egne objekter.

Objektorientering

Når jeg hadde fått testet nok med de enkle filene begynte jeg å skrive objekter. Jeg lagde en struktur som lignet litt på hvordan man gjør det i java, jeg hadde en klasse i en fil og mange filer. Det aller første jeg lagde var Window.py. Meningen var at dette skulle være objektet som holdt styr på hva som ble vist på skjermen. Etter det så skrev jeg også flere klasser som bare ble laget for å bli «inherited» fra, sånn som Veiw.py. Hele denne prosessen tok mye tid siden det var mange filer som skulle på plass og en komplisert struktur som måtte bygges.

Når jeg gjorde ferdig dette så hadde jeg mitt eget bibliotek med ulike komponenter jeg kunne bruke til å snakke over nett og vise ting på skjermen.

Nettverk og database

For at serveren skulle kommunisere med klienten måtte jeg skrive en server som hadde flere threads. Dette vil si at den kjørte ulik kode asynkront. Det gjør at jeg har en server som kan motta og sende data uten å bli påvirket av en klient som krasjer. Den er også koblet med databasen min, og laster opp data der når nye brukere kobler seg til. Databasen jeg bruker er mysql siden det virket praktisk og relasjonsbasert. Databasen min har tre tabeller som er knyttet sammen med relasjoner. Derfor mener jeg det er nyttig å bruke en relasjonsdatabase.

Server

Jeg satte opp virtuell maskin for å teste slik at jeg hadde et nettverk. På denne brukte jeg først windows server siden jeg er mest kjent med windows, men jeg gikk over til ubuntu server siden mange anbelafte linux i stedet. Dette gjorde også at det ble litt enklere å lage en step-by-step guide på hvordan sette opp serveren på en ny vm. Her bruker jeg da ssh for å koble meg opp fra klienten. Det er også mindre ressurskrevende å kjøre linux og det er det mest vanlige i virkeligheten.

Testing og dokumentasjon

Når jeg skrev en ny del med kode, så husket jeg alltid på å prøve den og sørge for at ting fungerte slik jeg ønsket det. Jeg gjorde mange tester for å sjekke at informasjonen sendt over nett kom frem på riktig måte, og jeg testet hvordan databasen tok imot informasjon.

Jeg har også skrevet mange kommentarer i koden min i form av «docstrings». Dette har jeg brukt til å lage en autogenerert api for prosjektet. Formateringen er ikke det beste, men den gir en god oversikt over alle filene og sammenhengen. Jeg har også skrevet opplæring til it-lærling og til sluttbruker, dette er veldig enkel forklaring av det man skal gjøre for å spille spillet. Jeg har også laget teknisk dokumentasjon for å holde oversikt over ulike maskiner og passord.

Sluttresultat

Spillet består av to deler; server koden og klient koden. De snakker sammen ved hjelp av det standard socketing biblioteket til python. Klienten bruker også pygame for å tegne opp ting på skjermen og for tastatur input. Alt som ligger under mappen «game» er klasser som jeg har skrevet selv som den bruker til å tegne opp ting på skjermen. Det som ligger under mappen «server» er separat kode som skal kjøre på en virtuell maskin. Den må også ha en mysql database som er satt opp riktig.

Erfaringer

Det var interessant å undersøke hvordan man bruker «socket.py» og jeg har lært mye om hvordan python fungerer som et kodespråk. Jeg har trent opp syntaxen min til et nivå hvor jeg skiller mellom bra og dårlig kode, og prøver alltid å gjøre ting på en bra måte.