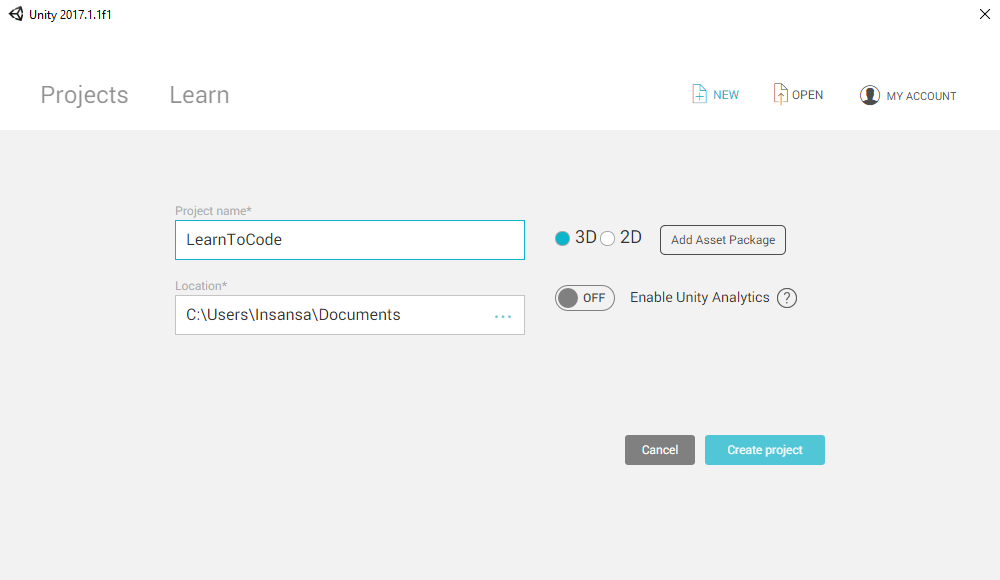
# Lær å kode i Unity – del 3.1: Variabler

I denne delen av kurset skal vi se nærmere på verdenen innenfor programmering. Programmering er en svært viktig del av det å lage et spill. Det er der du finner alle instruksene til hva som skal skje i spillet. Dette kan sammenliknes med for eksempel en matoppskrift:

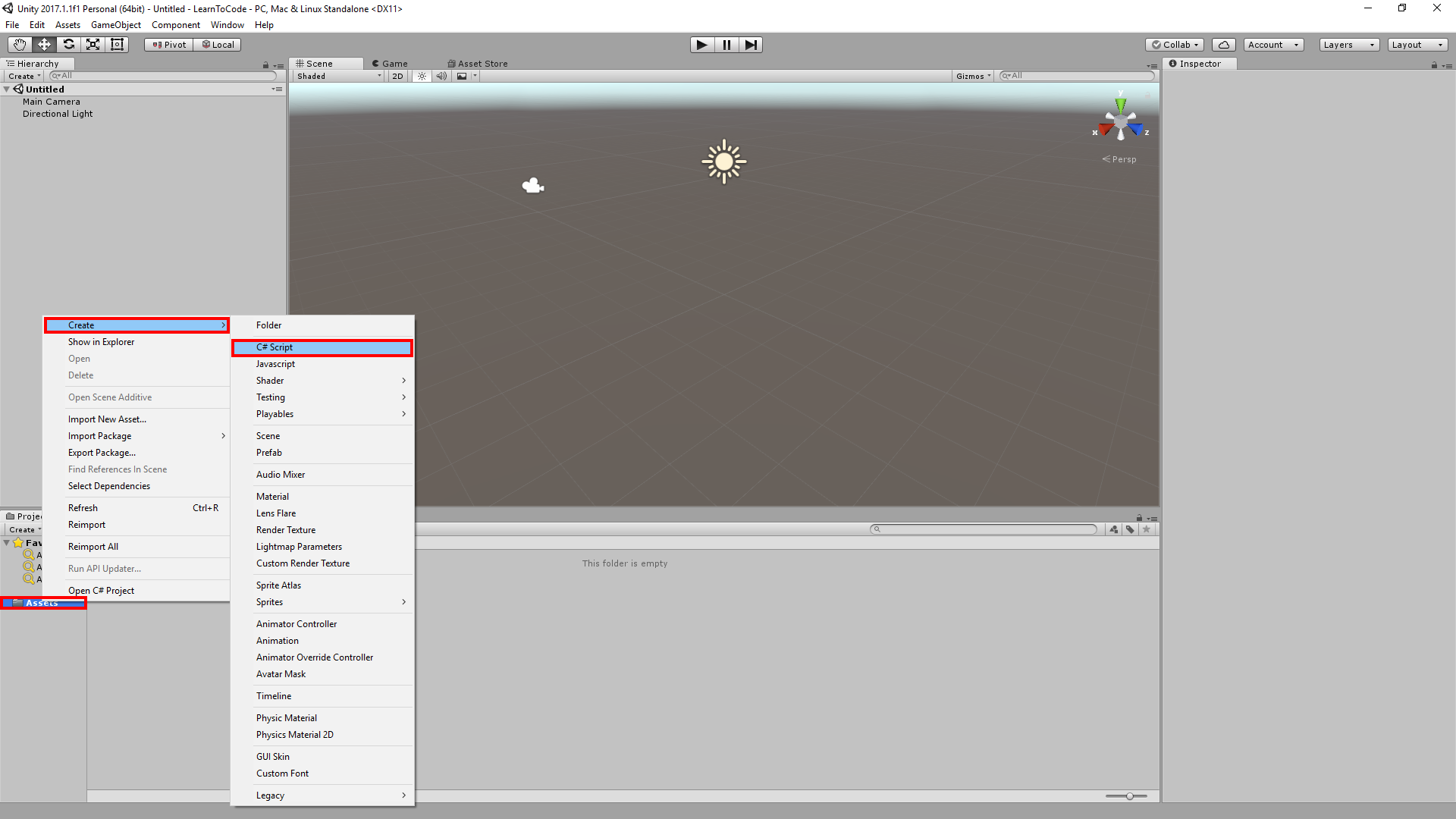


Du har en rekke med ingredienser, en instruksjon på hvordan ingrediensene skal brukes og til slutt får du et ferdig produkt. For at «oppskriften» skal kunne utføres må den være skrevet på et forståelig språk. For oss mennesker kan dette være norsk, engelsk osv. En datamaskin har egne språk den forstår. Det finnes tusenvis av slike ulike språk, og hver av dem har sine områder de er gode på. Språket vi skal bruke her heter C# (uttales C sharp) og brukes mye til spillprogrammering, spesielt i Unity.

Åpne Unity. Klikk file – new project og kall prosjektet for «LearnToCode».

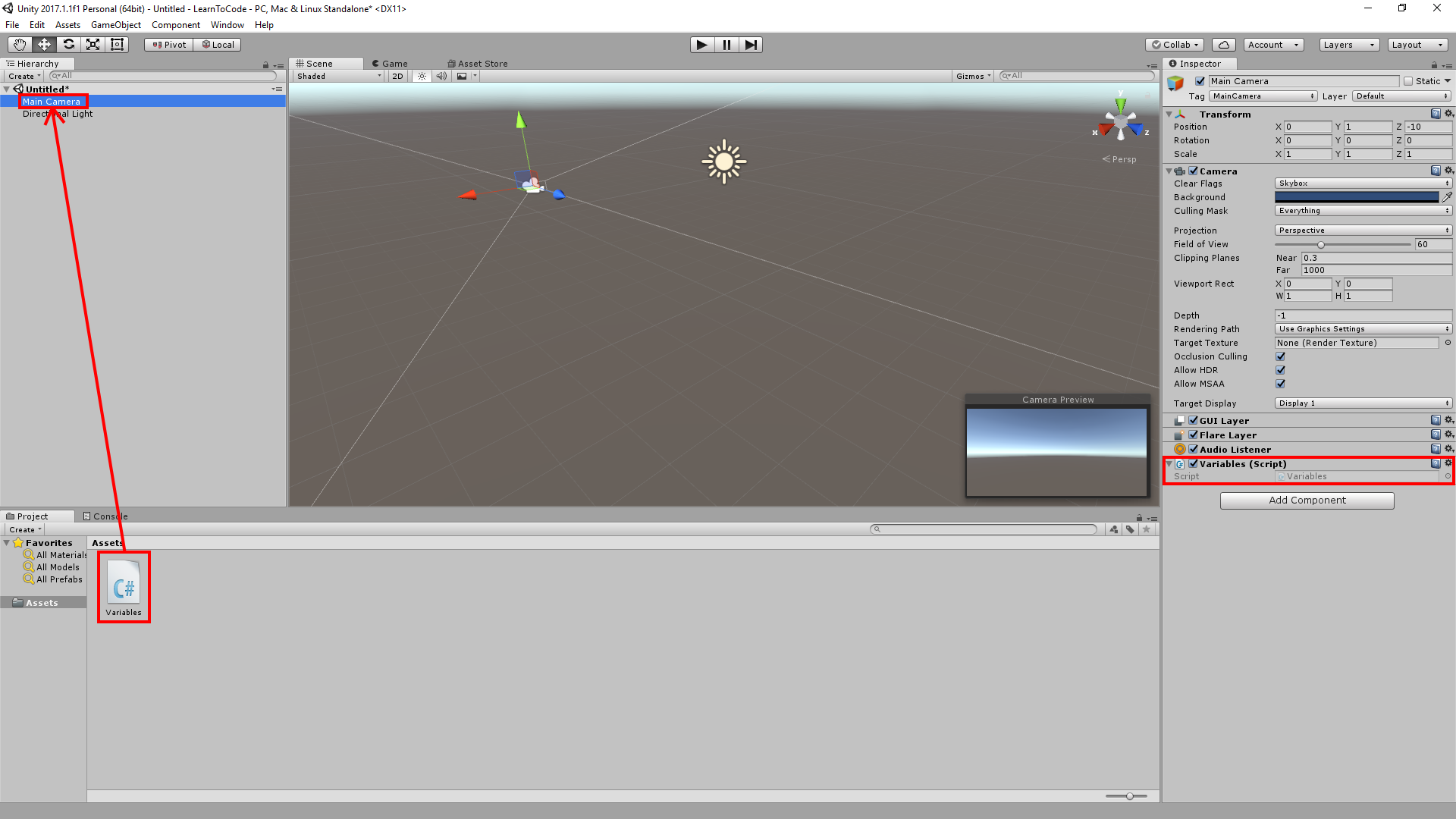


Klikk project – høyreklikk på assets – create – C# Script, kall den «Variables».

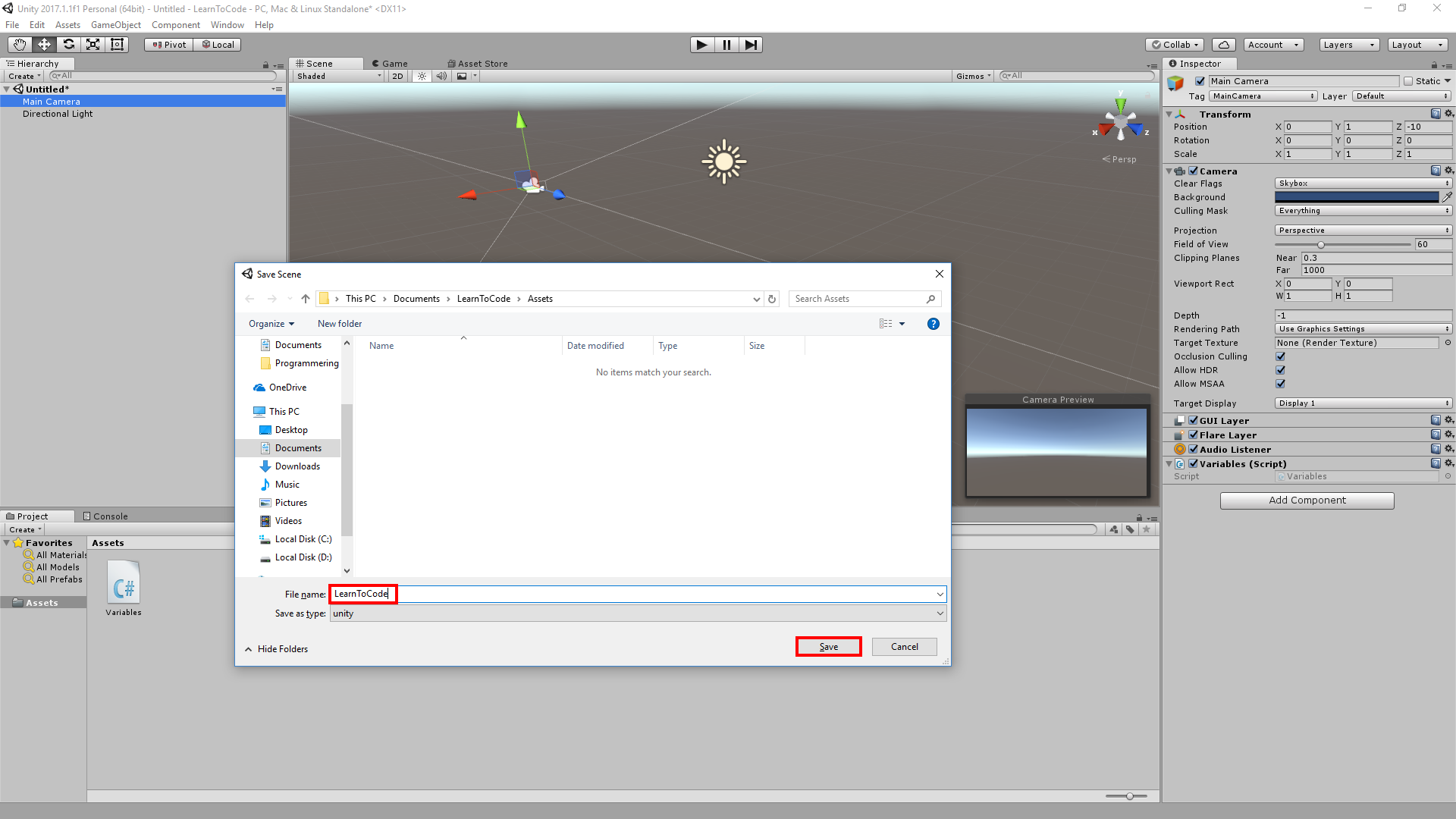


Når du jobber i Unity kommer du til å lage mange forskjellige «Scripts», og disse befinner seg inni prosjektet ditt slik at du lett kan dra dem over til objekter i spillet ditt. Så det første du skal gjøre er å dra skriptet ditt over til «Main Camera». Dette skriptet vil dermed starte når «Scene» starter. Konseptet med å feste kode til objekter er ulikt fra andre typer programmeringssystemer. Det er kun i Unity vi gjør dette, men selve kodedelen er den samme uansett hvilket program du bruker.

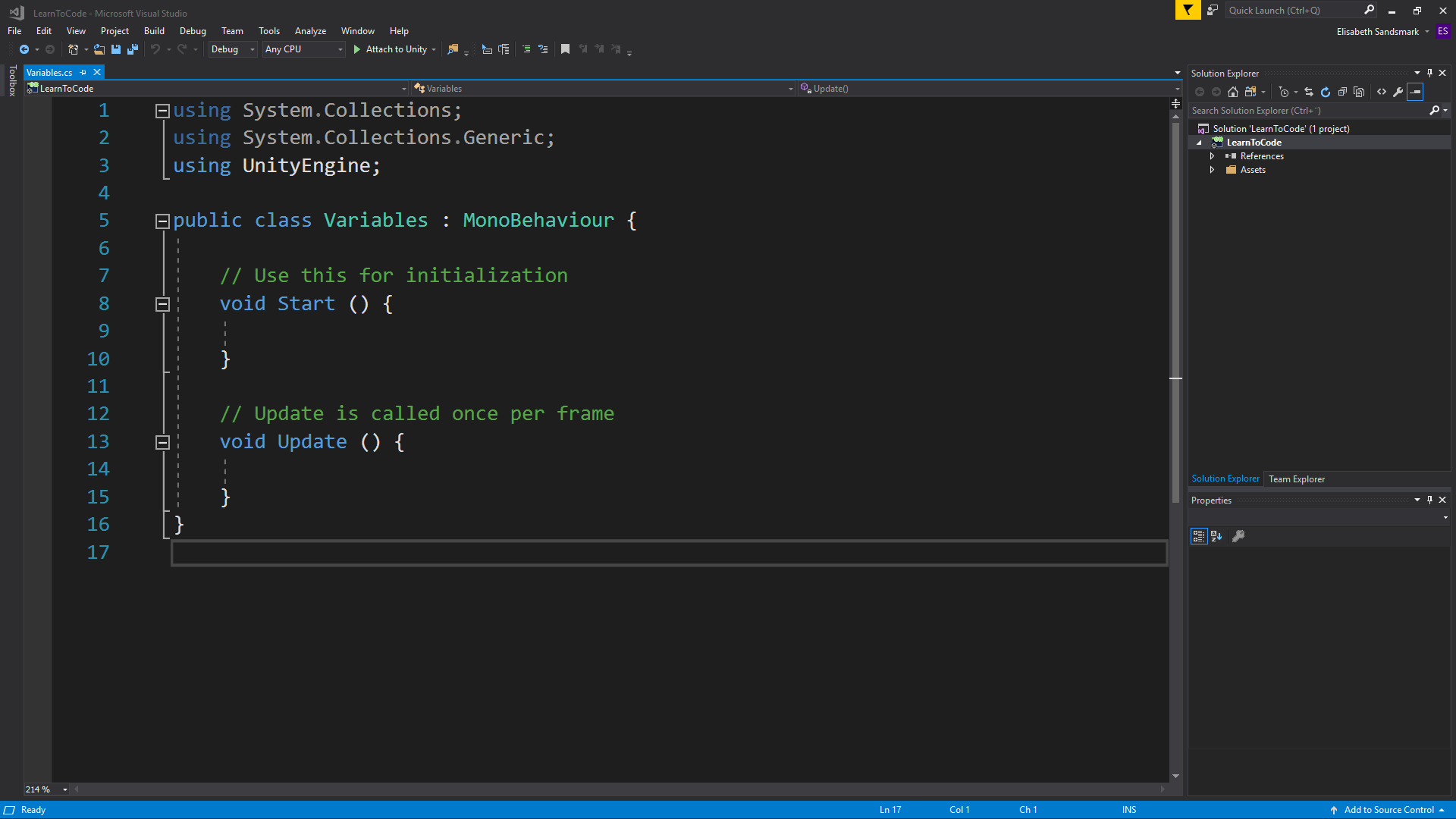
Hvis du nå klikker på Main Camera vil du kunne se skriptet ditt i «Inspector».



Lagre «Scene» (Ctrl + S) og kall den LearnToCode.



Åpne skriptet «Variables» (i MonoDevelop eller VisualStudio etc)



Legg merke til at det allerede befinner seg en «Start-funksjon» og en «Update-funksjon» i skriptet (mer om funksjoner senere). Start-funksjonen blir brukt når spillet starter, mens Update-funksjonen blir brukt om og om igjen gjennom hele spillet i en «loop»/sirkel.

## Variabler

Det første vi skal se på er variabler («variables»). Forestill deg at datamaskinens minne består av et stort antall celler (lagerplasser) hvor hver av disse cellene har sin egen adresse og kan lagre én enkelt verdi. For å reservere en av disse cellene må vi gi verdien et navn. Variabler er navngitte verdier som brukes for å lagre informasjon. Vi skiller mellom tallverdier (int) og tekst (String).

Dette er en visualisering av hvordan du kan tenke deg at det ser ut når informasjonen blir lagret, ikke slik det egentlig ser ut:

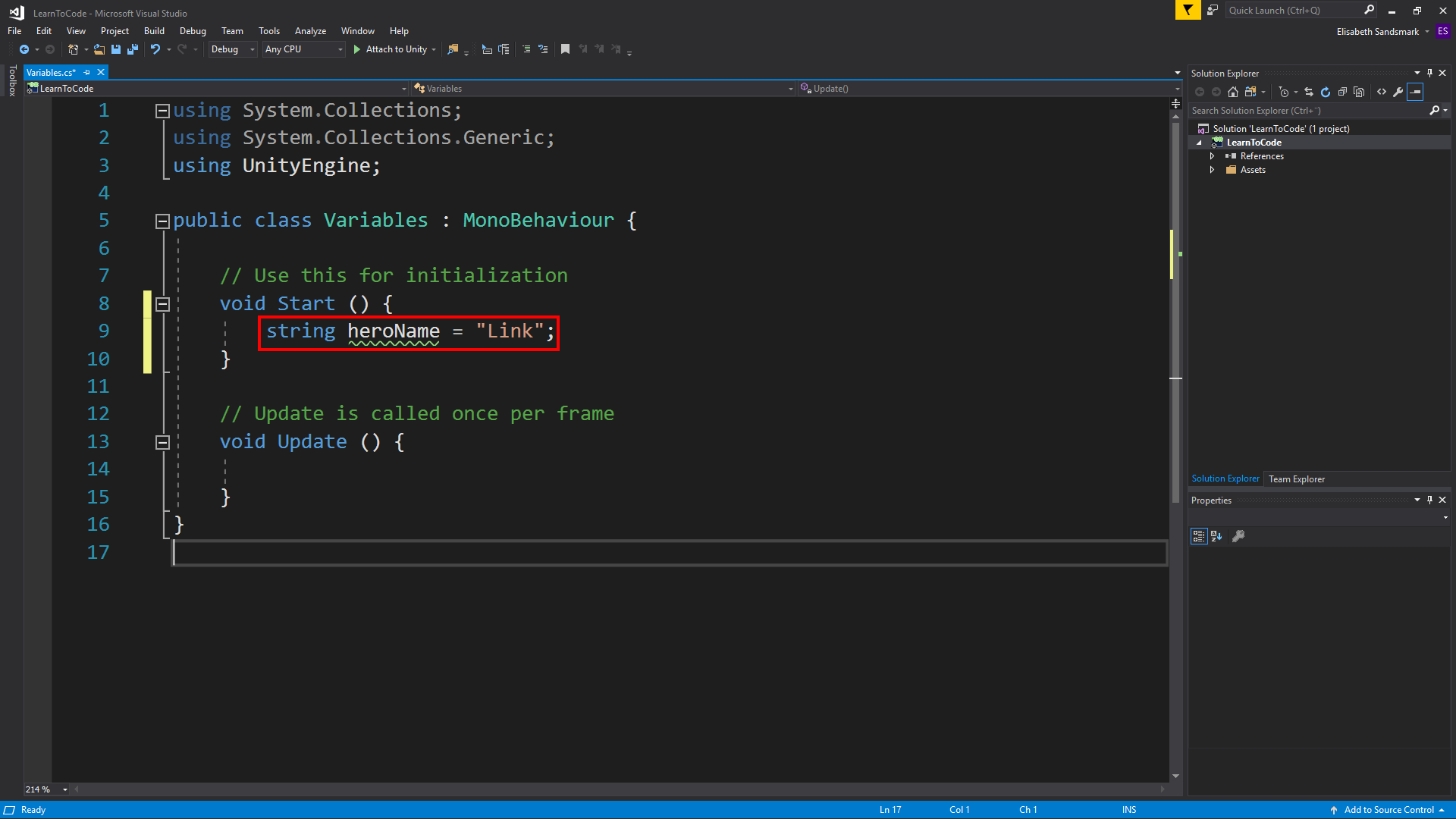
|  |  |  |
| --- | --- | --- |
| Adresse: 01011001  Variabel type: int  Variabelnavn: goldCarried  Verdi lagret: 32 | Adresse: 01011010  Variabel type: string  Variabelnavn: favoriteFruit  Verdi lagret: Pear | Adresse: 01011011  Variabel type: string  Variabelnavn: favoriteDrink  Verdi lagret: Milk |
| Adresse: 01011100  Variabel type: string  Variabelnavn: heroName  Verdi lagret: Link | Adresse: 01011101  Variabel type: string  Variabelnavn: equippedWeapon  Verdi lagret: Master Sword | Adresse: 01011110  Variabel type: int  Variabelnavn: heroAge  Verdi lagret: 15 |
| Adresse: 01011111  Variabel type: int  Variabelnavn: numberOfEyes  Verdi lagret: 2 | Adresse: 10100000  Variabel type: string  Variabelnavn: sidekickName  Verdi lagret: Bob | Adresse: 10100001  Variabel type: string  Variabelnavn: favoriteColor  Verdi lagret: green |

Når du jobber med et spill kommer du til å lage mange variabler og det kan være lurt å gi dem gode navn slik at du selv forstår hva variabelen peker mot. La oss starte med vår aller første variabel under Start-funksjonen. I et spill (for eksempel en RPG) kommer vi blant annet til å ha en hovedperson eller en helt. Denne helten trenger et navn. Kall variablen for heroName eller mainCharacter eller andre passende navn (så lenge det inneholder kun bokstaver kan du gi det hvilket som helst navn. Dette navnet er kun for deg som programmerer) og gi helten et navn f.eks. Link eller Superman osv. (det er dette navnet vi kommer til å se i spillet).

**string heroName = «Link»;**

Dette gjør at informasjonen om navnet til helten vår blir lagret i minnet til datamaskinen, og vi kan bruke denne informasjonen seinere i koden vår når vi måtte ønske.

Merk deg også at alle deklarasjoner (og for øvrig alle andre instruksjoner) blir avsluttet med et semikolon ;.



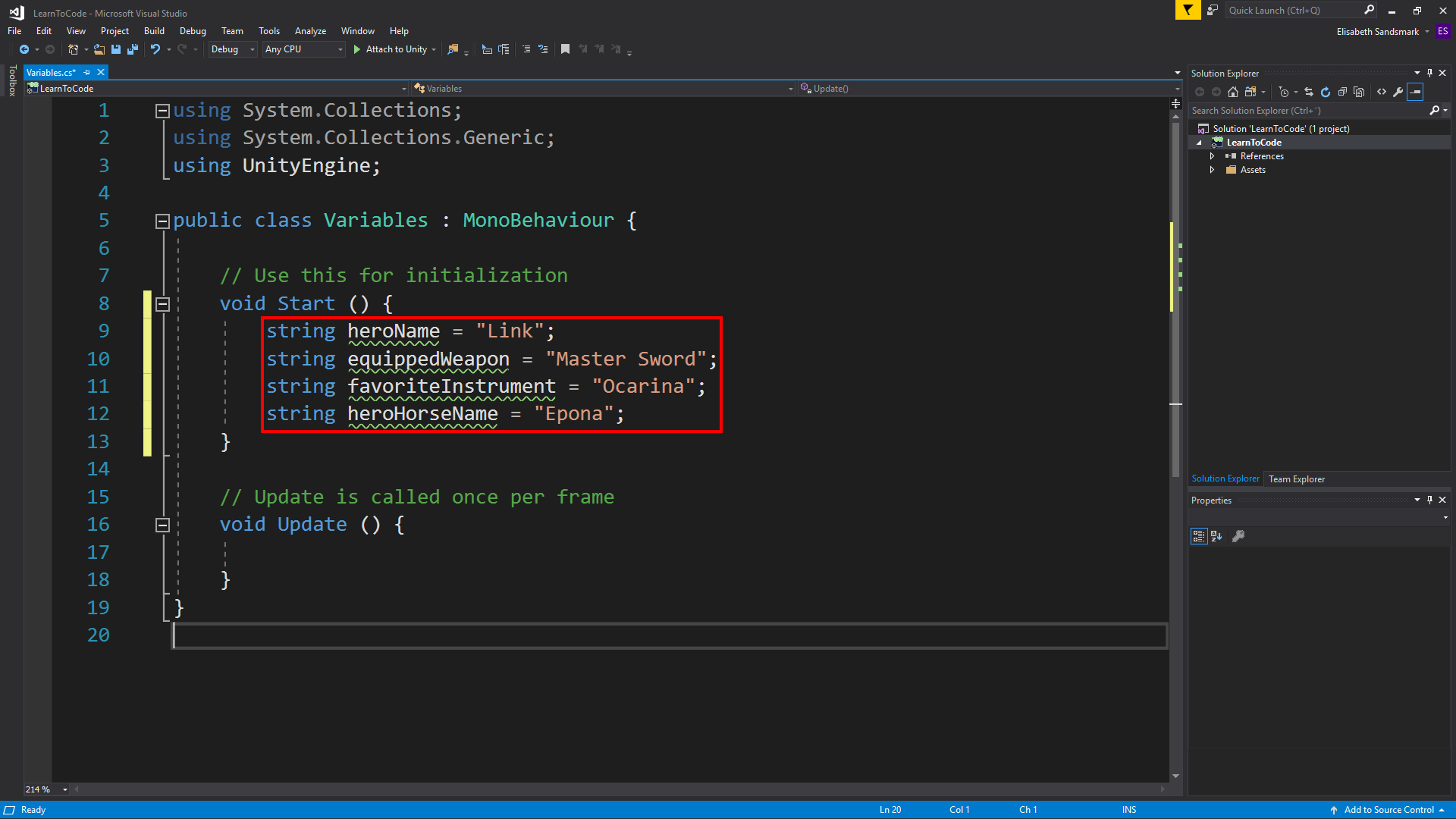
Neste vi kan lage er for eksempel et våpen til helten vår. (Gi dem hvilke som helst navn du ønsker)

**string equippedWeapon = «Master Sword»;**

Legg til så mange variabler du ønsker. Eksempler:

**string favoriteInstrument = «Ocarina»;**

**string heroHorseName = «Epona»;**



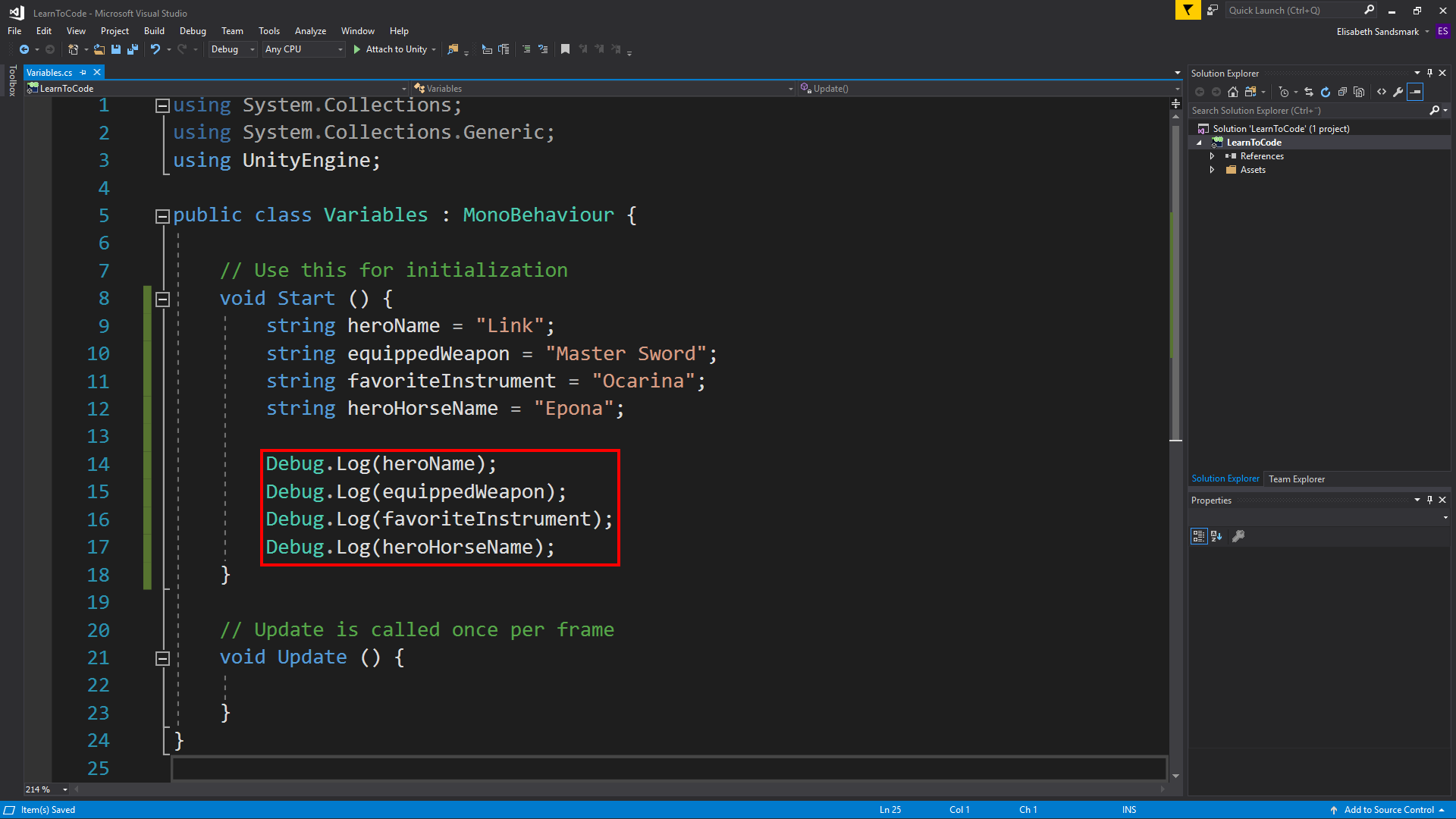
Vi kan nå for eksempel printe ut disse navnene til skjermen av spillet. Dette gjøres på følgende måte:

**Debug.Log (heroName);**

**Debug.Log (equippedWeapon);**

**Debug.Log (favoriteInstrument);**

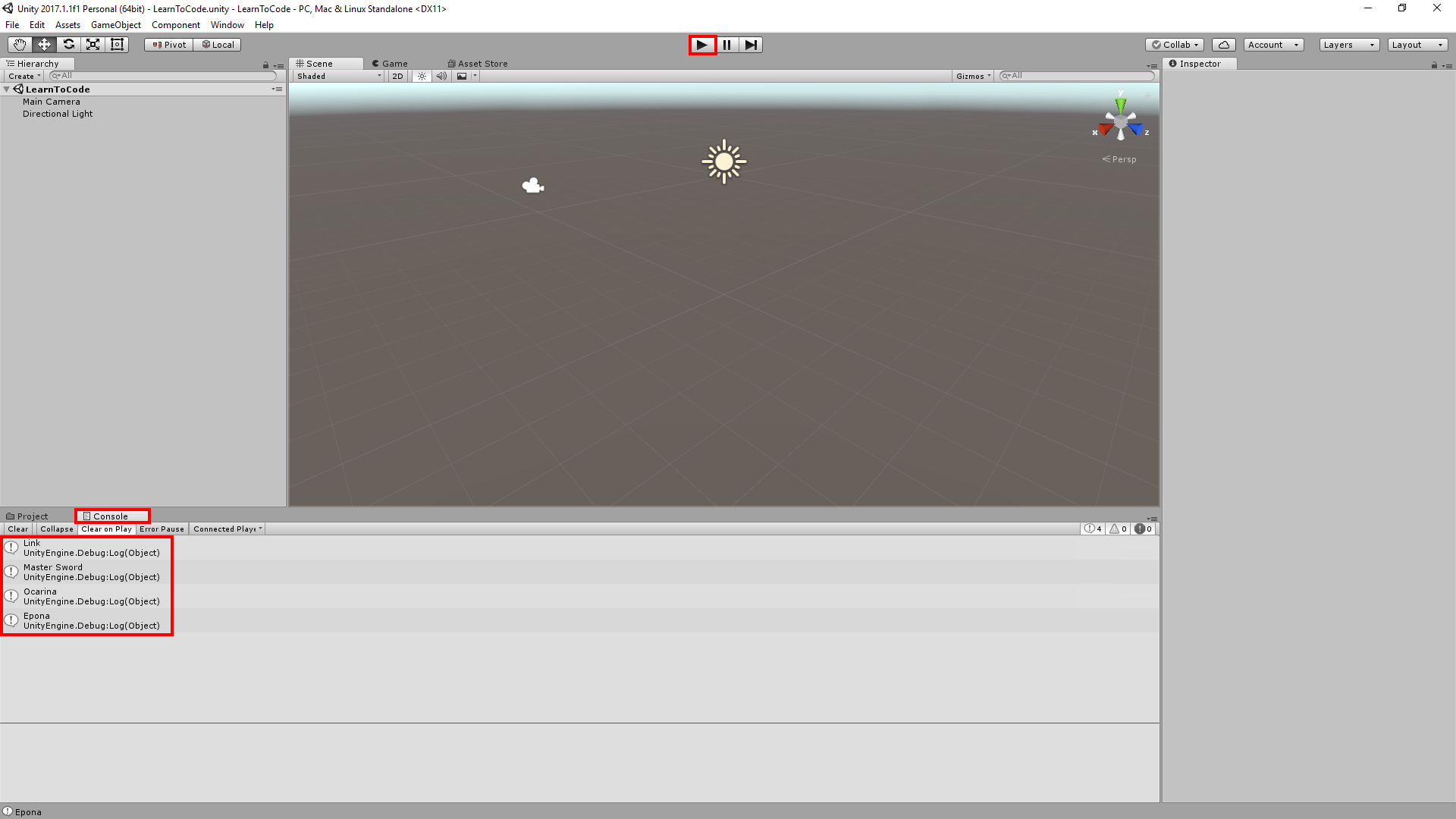
**Debug.Log (heroHorseName);**



Lagre dette og gå tilbake til Unity. Klikk på «Console» og klikk på play knappen for å kjøre spillet. (Klikk igjen for å avslutte kjøring av spillet.)

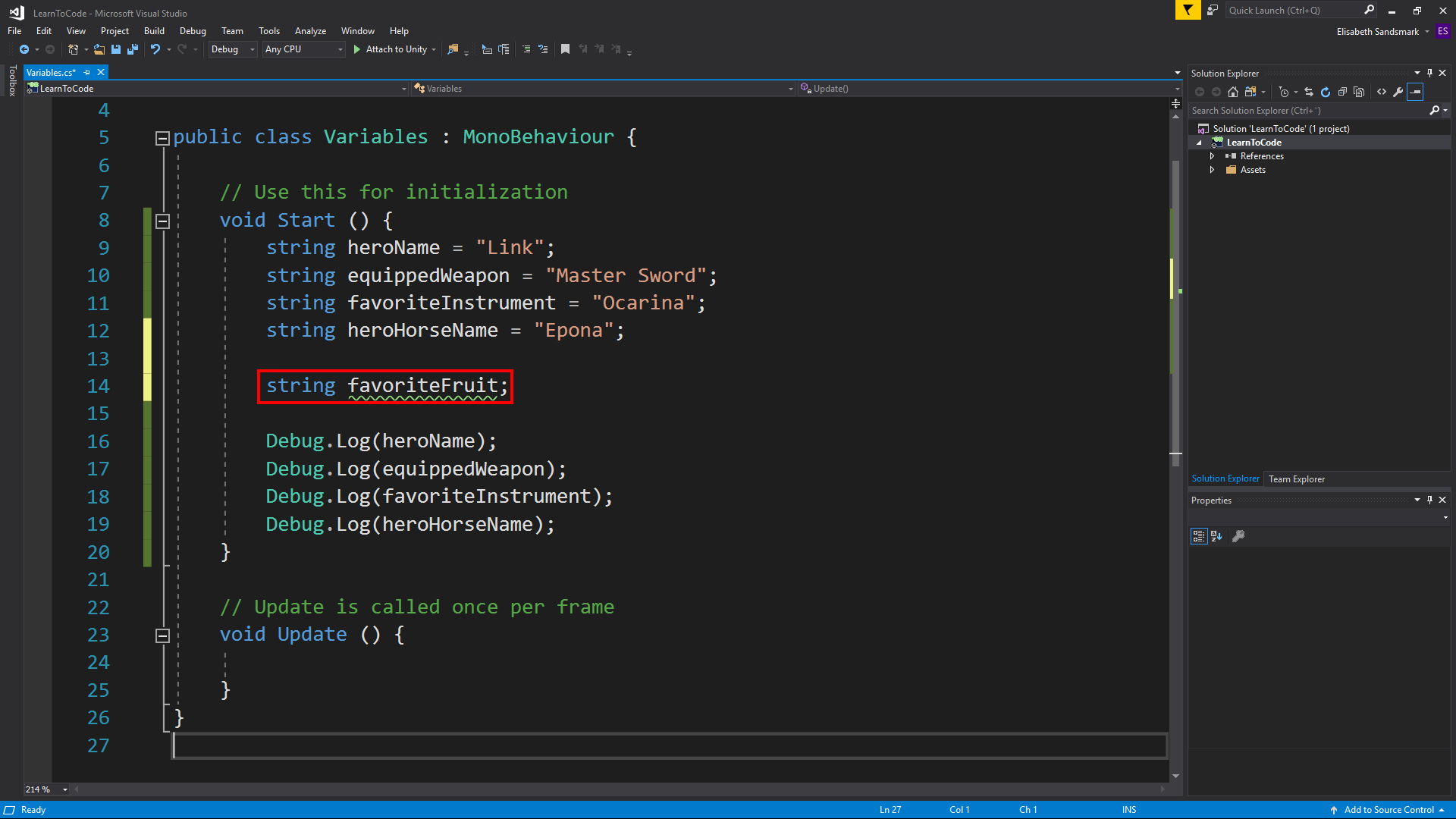
Legg merke til at det skrives ut alle variablene (Link, Master Sword, Ocarina og Epona dersom du ikke har laget dine egne variabler. Om du har laget dine egne variabler vil disse bli skrevet ut i stedet).

Kort oppsummert: Det du jobber med nå er data. Du spesifiserer en type (string), gir variablen et navn (eks heroName) og gir den så en verdi (eks. Link).



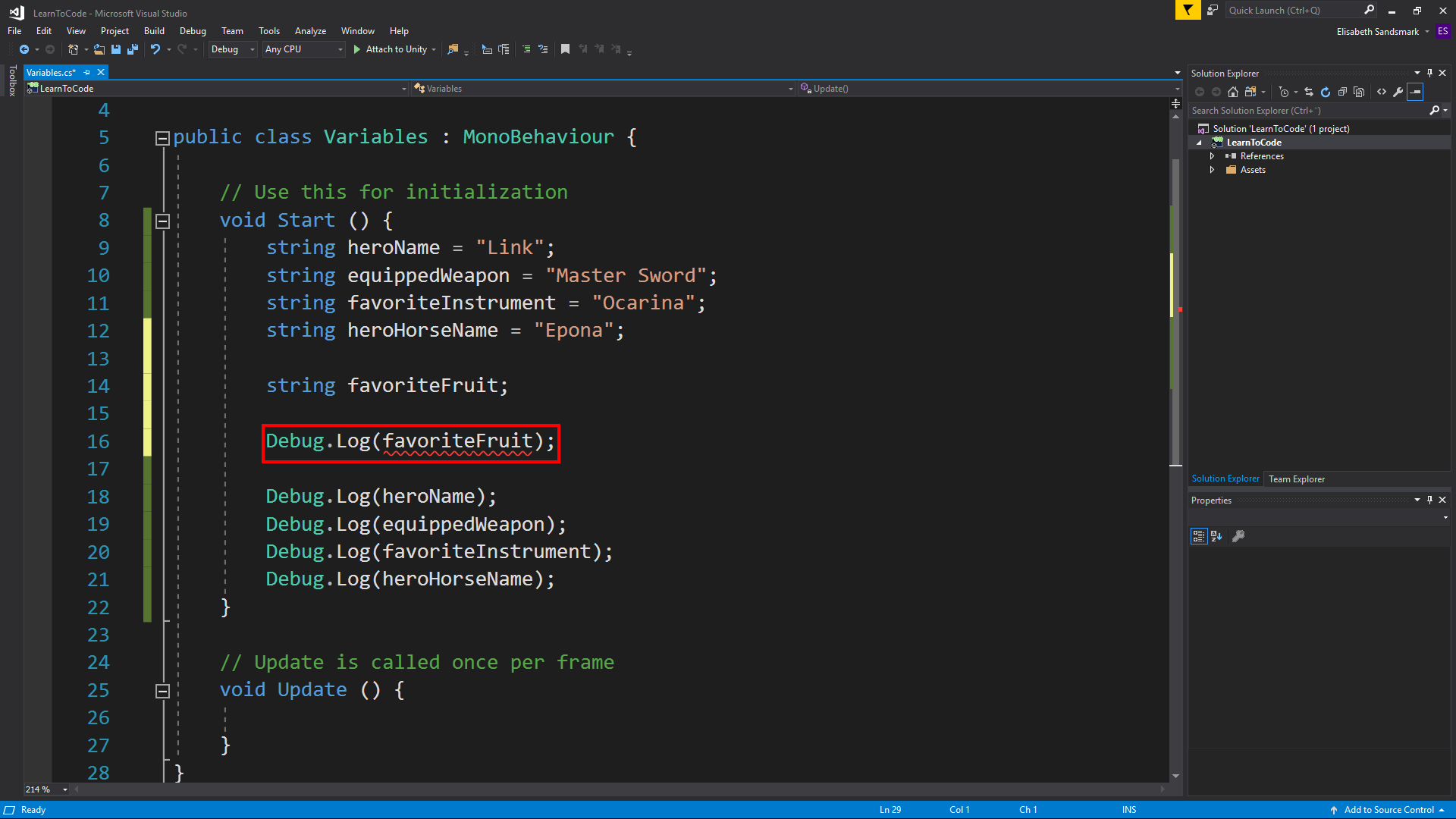
Du trenger ikke gi variablen en verdi med en gang. Kanskje har helten en favorittfrukt men vet ikke helt hvilken det er i starten. Eks:

**string favoriteFruit;**

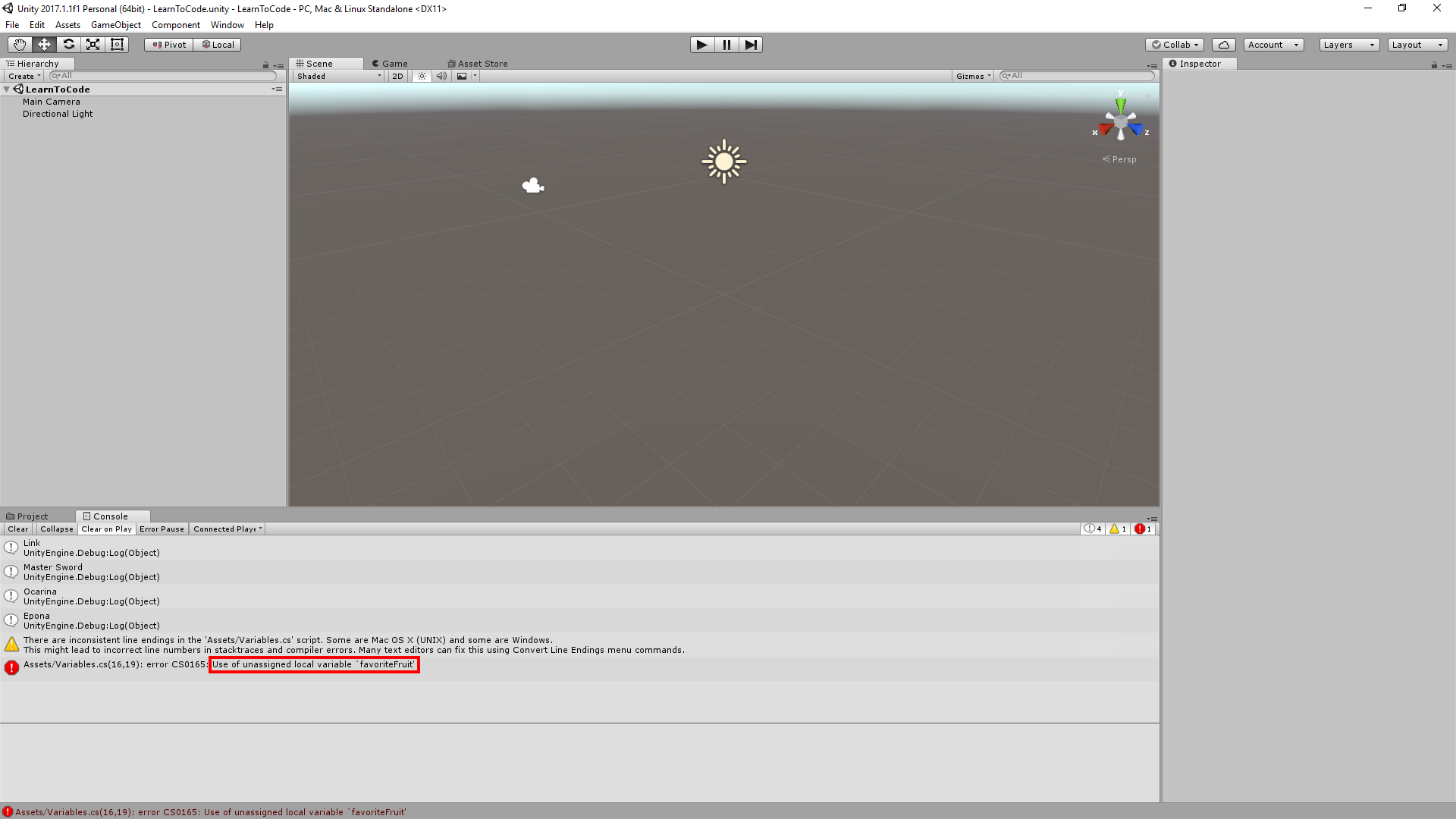


Du kan ikke hente denne informasjonen enda fordi vi ikke har gitt den en verdi. Hvis vi nå skriver:

**Debug.Log(favoriteFruit);**



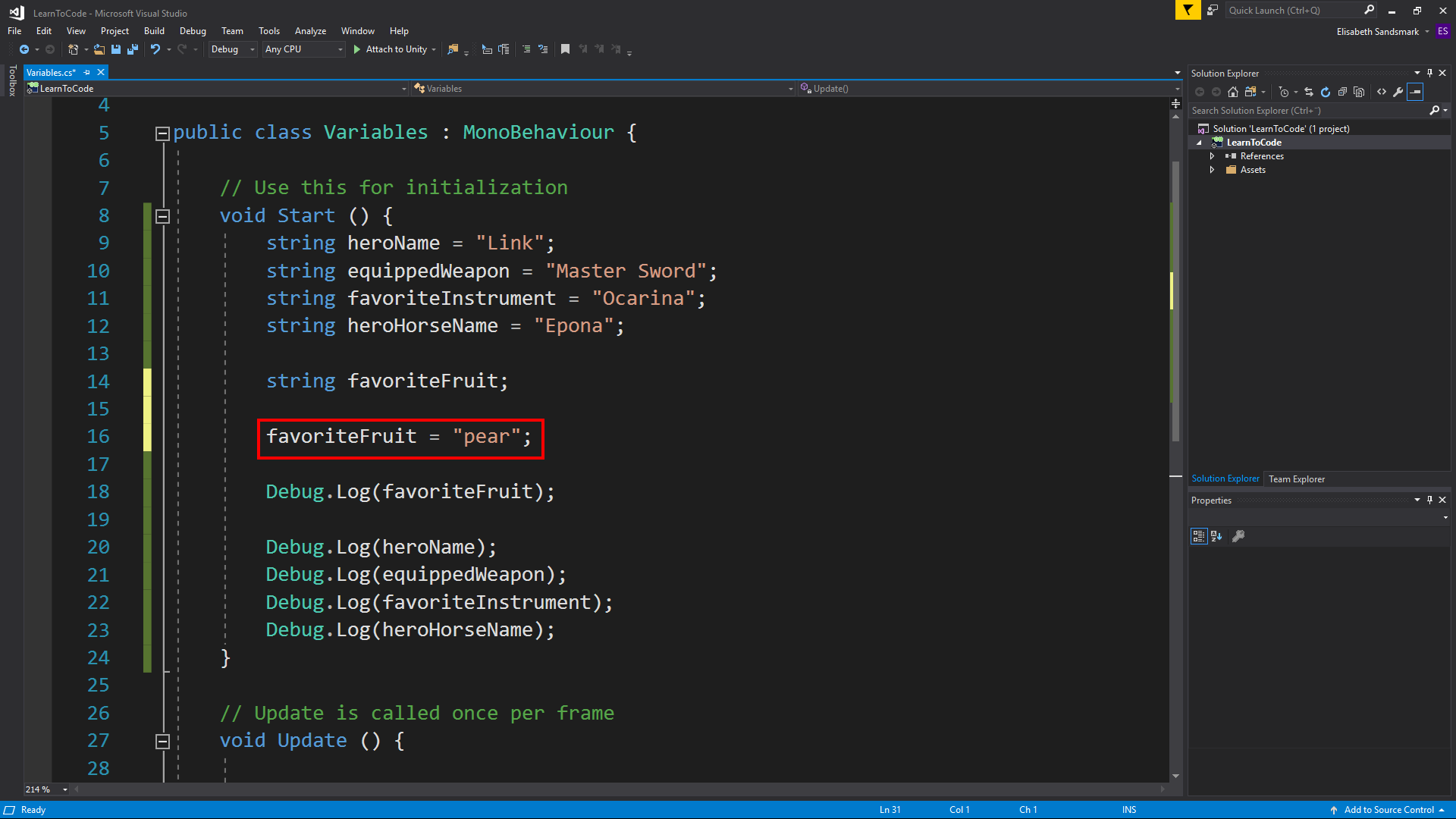
Vil vi få en feil «Use of unassigned local variable» dersom vi lagrer og prøver å kjøre programmet. Dersom du prøver å kjøre et spill som prøver å bruke en variabel som ikke har en verdi, vil spillet mest sannsynlig krasje eller ikke kjøre i det hele tatt. Maskinen blir forvirret fordi den leter etter en verdi som ikke eksisterer.



La oss si at vi seinere finner ut at denne favorittfrukten er pære (OBS: pass på å ikke bruke norske bokstaver eller andre tegn som ikke er i det engelske alfabetet a-z. Maskinen forstår som regel ikke hva du prøver å si og vi får feil når vi prøver å kjøre programmet).

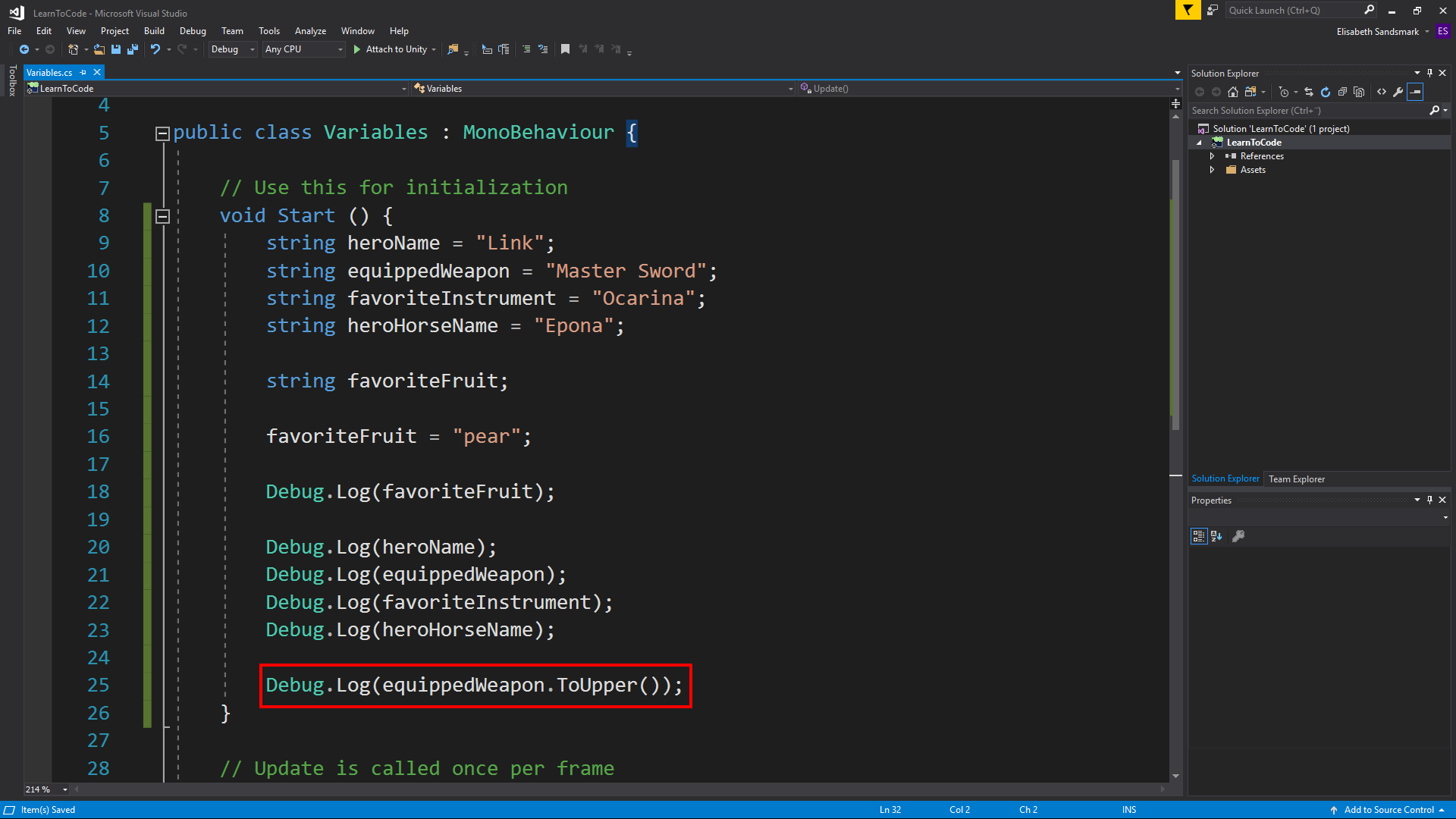
**favoriteFruit = «pear»;**

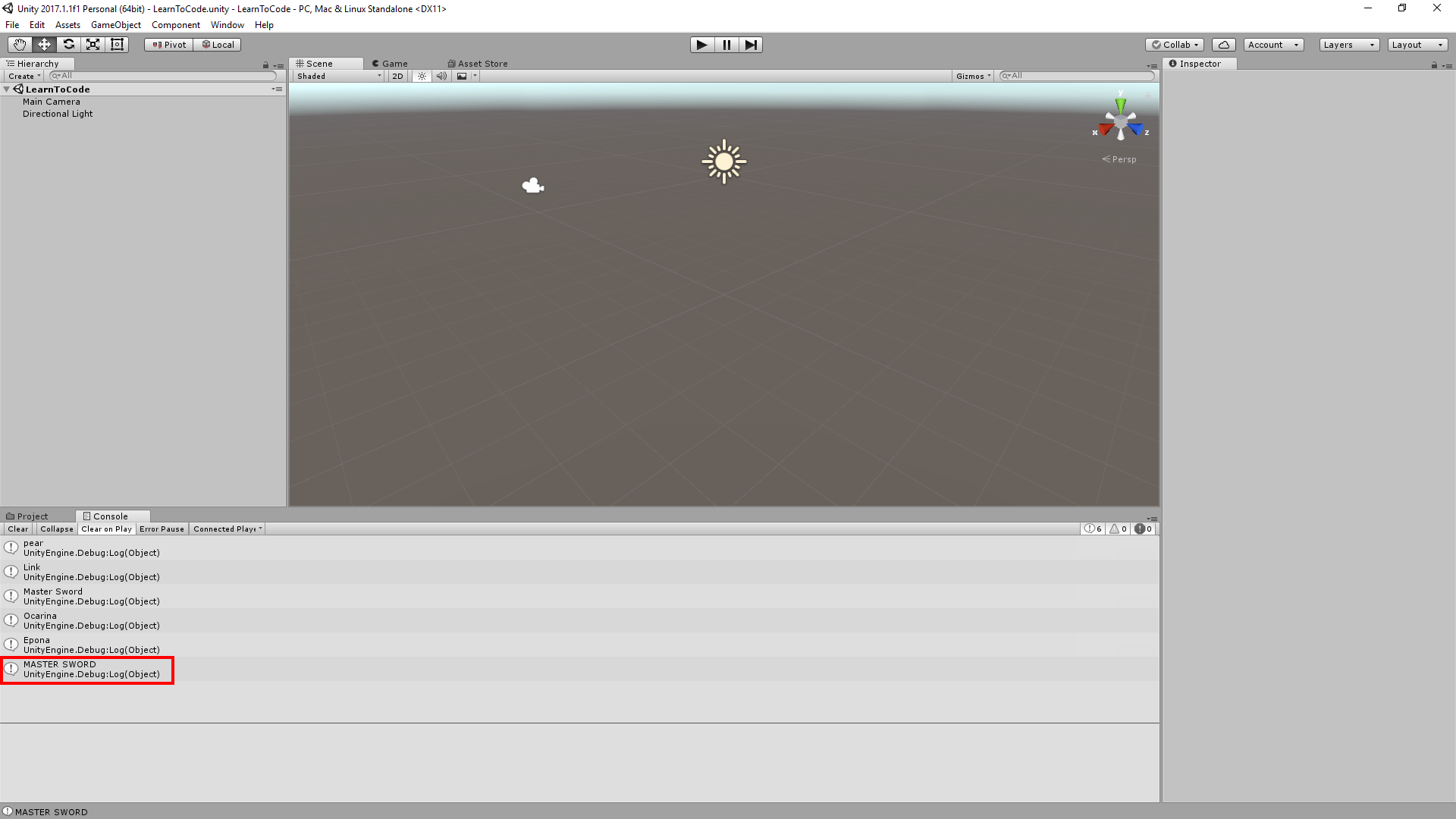
Dette må stå før *Debug.Log(favoriteFruit);* for at programmet skal kunne registrere at variablen faktisk har en verdi. Når maskinen leser kode, så leser den fra topp til bunn (linje 1 til linje x) med mindre vi spesifiserer noe annet underveis. Lagre og test at alt fungerer som det skal.



Det er mye vi kan gjøre med strings. For å ta et eksempel kan vi blant annet gjøre hele teksten i en string om til store bokstaver:

**Debug.Log(equippedWeapon.ToUpper ());**





Alt etter . (eks ToUpper, IndexOf, GetType) er forskjellige funksjoner du kan kalle på og bruke for å modifisere dine variabler. Prøv deg frem og sjekk på nettet hva de ulike funksjonene gjør. (Søk f.eks. på C# string ToUpper).

Full kode til «Variables» (dersom du får problemer underveis):

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Variables : MonoBehaviour {

// Use this for initialization

void Start () {

string heroName = "Link";

string equippedWeapon = "Master Sword";

string favoriteInstrument = "Ocarina";

string heroHorseName = "Epona";

string favoriteFruit;

favoriteFruit = "pear";

Debug.Log(favoriteFruit);

Debug.Log(heroName);

Debug.Log(equippedWeapon);

Debug.Log(favoriteInstrument);

Debug.Log(heroHorseName);

Debug.Log(equippedWeapon.ToUpper());

}

// Update is called once per frame

void Update () {

}

}