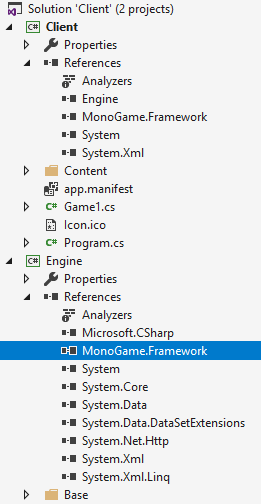
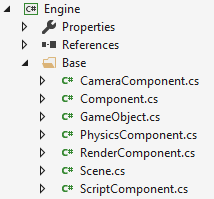
# Week 4.2

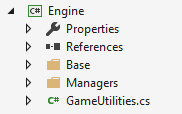
* Create a new MonoGame Project (Windows Desktop DirectX) called Client
* Add a new Class Library to the solution called Engine
  + Set the target .NET Framework version to be 4.5
  + Add a reference to MonoGame
  + C:\Program Files (x86)\MonoGame\v3.0\Assemblies\Windows\MonoGame.Framework.dll
* In the WindowsClient project add a reference to the Engine project



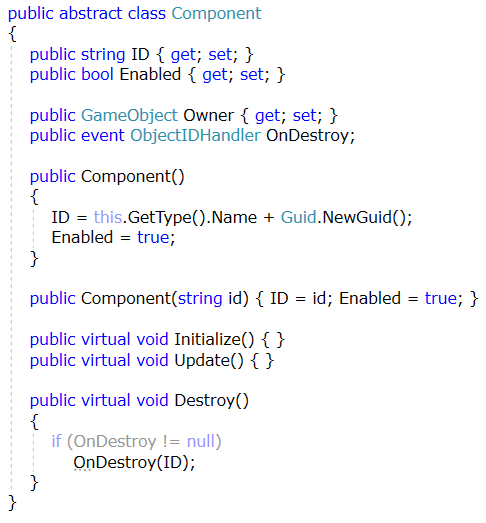
* In the Engine project create a new folder called Base
* Create 7 new classes as shown below



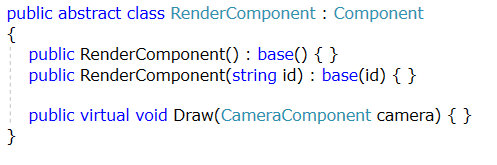
## Game Utilities



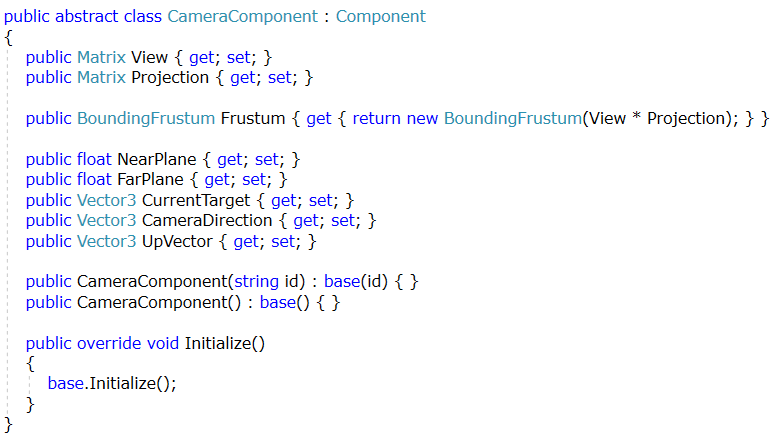
# Component



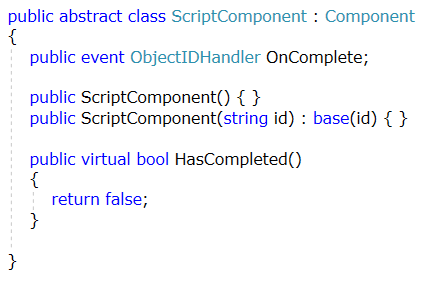
# Render Component



# Camera Component



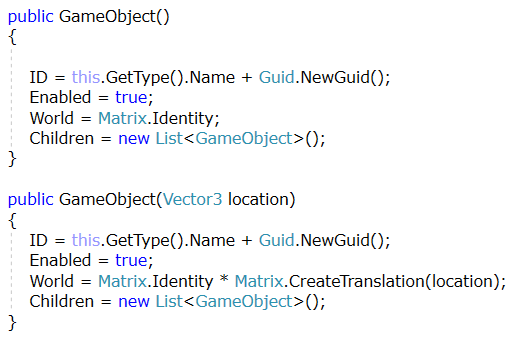
# Script Component



# Game Object

## Fields and Properties





## Methods



# Functional Requirements

## Initialize

* Call the initialize method for every component in the components collection
* Set the initialized of game object to be true

## Add Component

* Set the Owner property of the component to the current game object
* If the game object has already been initialized, call initialize on the new component immediately
* Assign an event handler to the OnDestroy event of the component (+= TAB TAB)
* Add the component to the component collection

## Component\_OnDestroy Event Handler

* Add the ID of the component to the awaiting removal list

## Remove Component (int)

* Remove the component at the given index in the components collection

## Remove Component (String)

* Remove the component with given ID from the components collection

## Remove Component (Component)

* Remove the given component from the components collection

## Destroy

* Clear the components collection
* Raise the OnDestroy Event

## Update

* Call the update method for every component that is enabled in the components collection
* For every string the awaiting removal list
  + Call the remove component method, passing the string ID of the component

## Draw (Camera Component)

* For every Render Component in the components list
  + If the component is enabled, call Draw on the component

## Get Distance TO (Game Object)

* Return the distance between the current game object and the game object argument

## Has Component <T>

* Generic method
* Return a Boolean indicating whether the components collection contains a component oif the matching the specified type
  + GetType() and GetType().IsSubclassOf()

## Get Component (string)

* Return the component with the matching ID

## Get Component (Type)

* Return the first component with matching type

## Get Components (Type)

* Return a list of all components with the matching type

## Get Component (T)

* Generic method
* Return the first component with matching type

## Get Components (T)

* Generic method
* Return a list of all components with the matching type