Classes1, part 2

1. Content

2. Objectives

At the end of these exercises, you should be able to:

- Define an animated sprite class.
- Convert an existing class-less game into a class game.

We advise you to **make your own summary of topics** that are new to you.

3. Exercises

Your name, first name and group should be mentioned at the top of each cpp file.

Give your **projects** the same **name** as mentioned in the title of the exercise, e.g. **ClassesBasics**. Other names will be rejected.

3.1. Arrays containing object pointers (TestArrays)

Create a new console project with name **TestArrays** in your **1DAExx_11_name_firstname** folder.

Let's do some tests on arrays of pointers to objects. Declare and define a function **TestArrays** and call it in the main function. Then write the code of following exercise in the definition of the **TestArrays** function.

a. An array of object pointers

In previous lab we made a Time class. Copy that class into this project.

Define an array with 4 elements of type **pointer to Time object** (Time*) using default uniform initialization in the main function. Use a constant for the array size!

Use a for loop to create 4 Time objects using dynamic memory allocation and using random values for the seconds and save the addresses of the created objects in these array elements.

Build, run and verify the content of the objects using a breakpoint.

Change the seconds of these Time objects using their AddSeconds method.

Build, run and verify the content of the objects using a breakpoint.

Don't forget to delete these objects before leaving the function.

3.2. TexturesAndSprites

Create a new project with name **TexturesAndSprites** in your **1DAExx_11_name_firstname** folder.

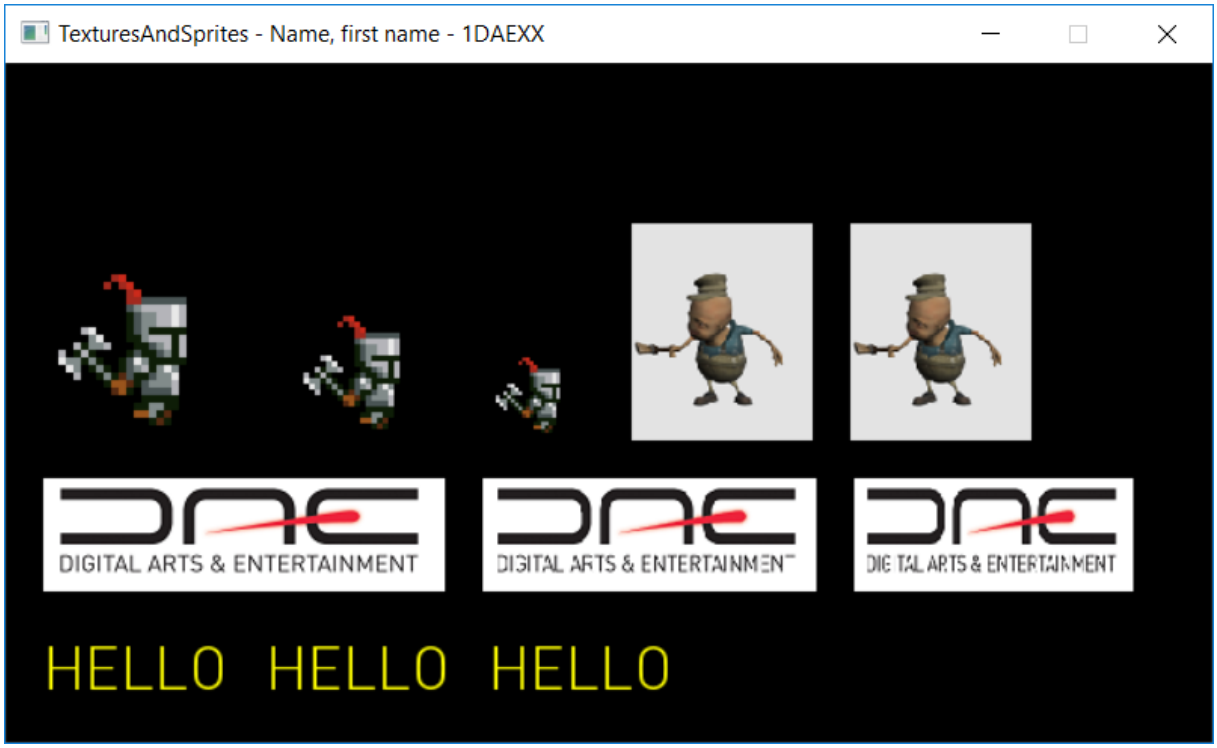
Delete the generated file **TexturesAndSprites.cpp.**

Use the **Functions Framework** and allow to overwrite **pch.h**

Adapt the window title in main.cpp.

Create a **Sprite** class and use it to create and draw the running knight and the Tibo sprite.

At the end of these exercises, you should have a window that looks like this.



3.2.1. Create and draw some Texture objects

Add 2 Textures, one for the DAE image and the other for some text.

Then draw the text-texture 3 times on the window.

Also draw the DAE-texture 3 times using a destination rectangle with decreasing width.

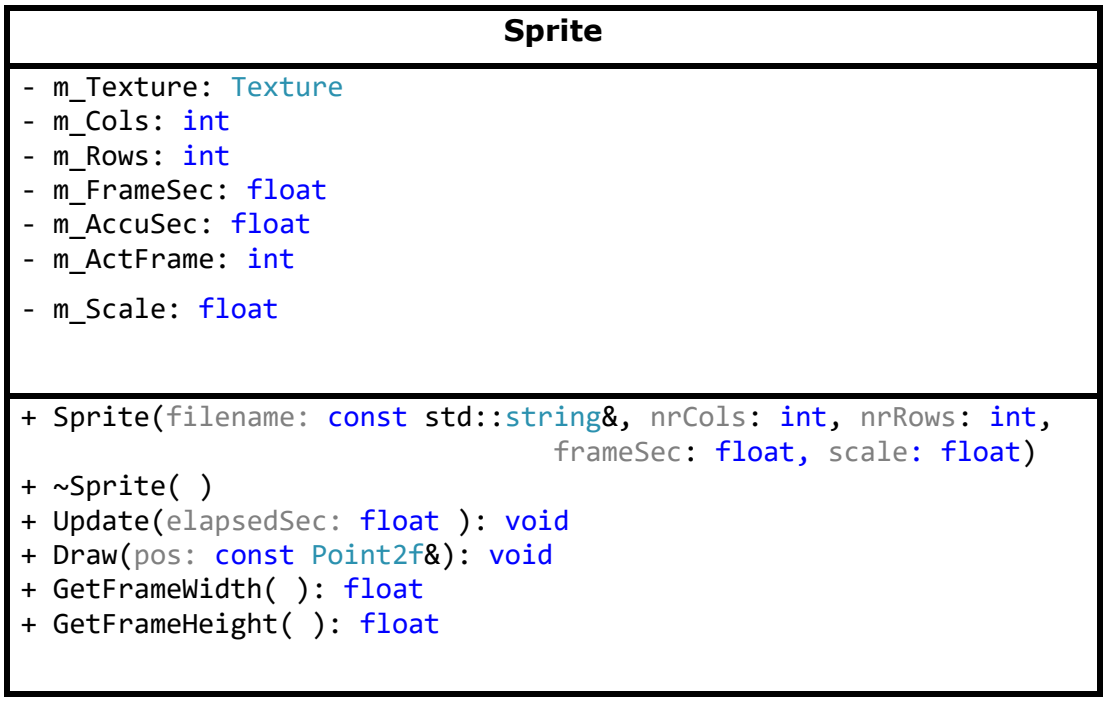
Also test some possible error cases, verify what happens when a wrong file was specified (wrong path).

3.2.2. Sprite class

In a previous lab you created a Sprite struct. Let's define a Sprite class, it will contain more or less the same data members, however also some additional functionality. The goal of this class is to draw animated images on the window.

a. Define the class

This is the UML diagram.



Here some extra info regarding the functionality of this Sprite class:

- The **constructor**: loads a Texture for the given image and initializes the member variables.
- **Destructor**: Does this class need one? Tip: do not add a destructor if it would be an empty one (see later).
- **Update**: calculates whether it is time for the next frame as indicated by m_AccuSec, m_FrameSec and m_ActFrame (see a previous lab about sprites)
- **Draw**: Draws the part of the texture as indicated by the property m_ActFrame at position *pos* on the window. The *m_scale* member variable defines the desired scaling.

b. Test the class

Make an array to hold 5 Sprite pointers.

Make 3 heap-objects of this class for the knight sprite and store their pointers in the array. Each knight uses a different scale.

Make 2 heap-objects of this class for the tibo sprite and also store their pointers in the array.

Draw the knights next to each other.

Draw the Tibo sprites.

3.3. Refactor

Take the exercises of previous lab, and change the code so that arrays of pointers are used, wherever is seems useful.

3.4. 2D based Grid game

Make a copy of your 2D based grid game project in the *1DAExx_11_name_firstname* folder.

Redesign it in such a way that it uses classes wherever this it appropriate.

4. Submission instructions

You have to upload the folder *1DAExx_11_name_firstname, however first clean up each project. Perform the steps below for each project in this folder:*

- In Solution Explorer: Select the solution, RMB, choose **Clean Solution**.
- Then **close** the project in Visual Studio.
- Delete the .vs folder.

Compress this *1DAExx_11_name_firstname* folder and upload it before the start of the first lab next week.

5. References