

# Mobiiliohjelmointi/Mobile programming Assignment

**Topic - MobiPonged**

**The students are required to implement a game, MobiPonged, on the Symbian platform.**

**The program must follow the Symbian architecture which separates the game logic (implemented as a DLL-project) and the User Interface into 2 different projects.**

**The game should be a single person game with good computer artificial intelligence (AI).**

**For an additional point, at the students implement an additional feature, of their own choosing to the game**

## **Requirements**

Groups of 2-3.

## **Grading.**

The project will be graded on a scale from 0-2, which will be added as bonus points to your exam. These are awarded as follows

0 – Project is accepted after significant corrections are made.

1 – Project work is accepted, no corrections required.

2 – Awarded at the discretion of the local assistants for the implementation of an additional feature

## **Environment**

It is recommended that students use Carbide.c++ ver 1.2 and either SDK 2<sup>nd</sup> edition FP3/ SDK 3<sup>rd</sup> Edition FP1. Students can use other environments, but should be aware that they will need to read additional material and time in order to complete the task.

## **Description**

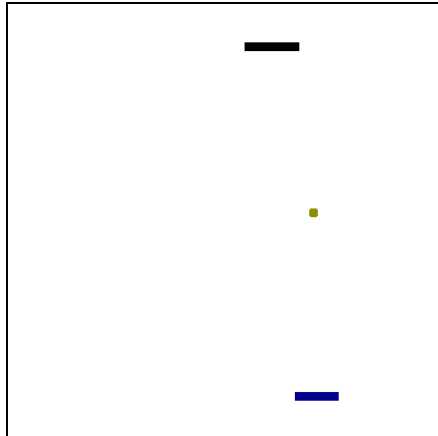
MobiPonged is a game based on table tennis. A history of MobiPonged can be found at <http://en.wikipedia.org/wiki/Pong>

An example of game play can be found here and is shown in Figure 1.

<http://www.falstad.com/pong/>

An example of a C code logic can be found here

<http://www.cppgameprogramming.com/cgi/nav.cgi?page=pong>



**Figure 1. basic MobiPonged Game. With one ball and two paddles.**

### **User Interface**

The User Interface must have sufficient space to play the game. In addition it must have options that allow the player, to start the game, to stop the game, restart a paused game, reset the game, show the current score of the game and to view previous highest scores of the game.

### **Game Logic:**

The game logic is implemented in a separate DLL. Examples of code logic can be found on the internet. These code segments have to be correctly ported to the Symbian environment.

Previous high scores for the game need to be stored in a file which is loaded into memory when a game is started. The highest scores are shown in a dialog. Up to 5 previous scores should be kept, including the dates when those scores were achieved.

### **Guidelines.**

The students should follow the recommended Symbian coding conventions. Students may not copy code between groups (discussion is ok). The assistant will be particularly focused on trying to determine if the students actually did the work themselves by asking questions during the submission sessions.

### **Submissions:**

The assignment has two phases of submission and a final submission.

Submission 1 : Students must have familiarized themselves with the tools, created two projects: engine (DLL) and UI. Additionally they must have linked them together. At the first submission, this is demonstrated by calling some method of the engine.

Submission 2 : Students must continue with the previous work, showing that the MobiPonged paddles and ball can actually move.

Submission (final) : Students must provide a working model, which should show that it is possible to release the code. The group demonstrates their application. The assistant checks the students work and asks questions where appropriate. After this inspection, the

students must submit the final program by email for inspection. Further specific instructions should be available from your local university web resources.

**Extra Features**

For an extra point, the students can add an extra feature to the game. This is an opportunity for the students to get creative. It is not necessarily related to the amount of coding/work involved, but whether the students can come up with something interesting/unique that adds value or interest to the game.