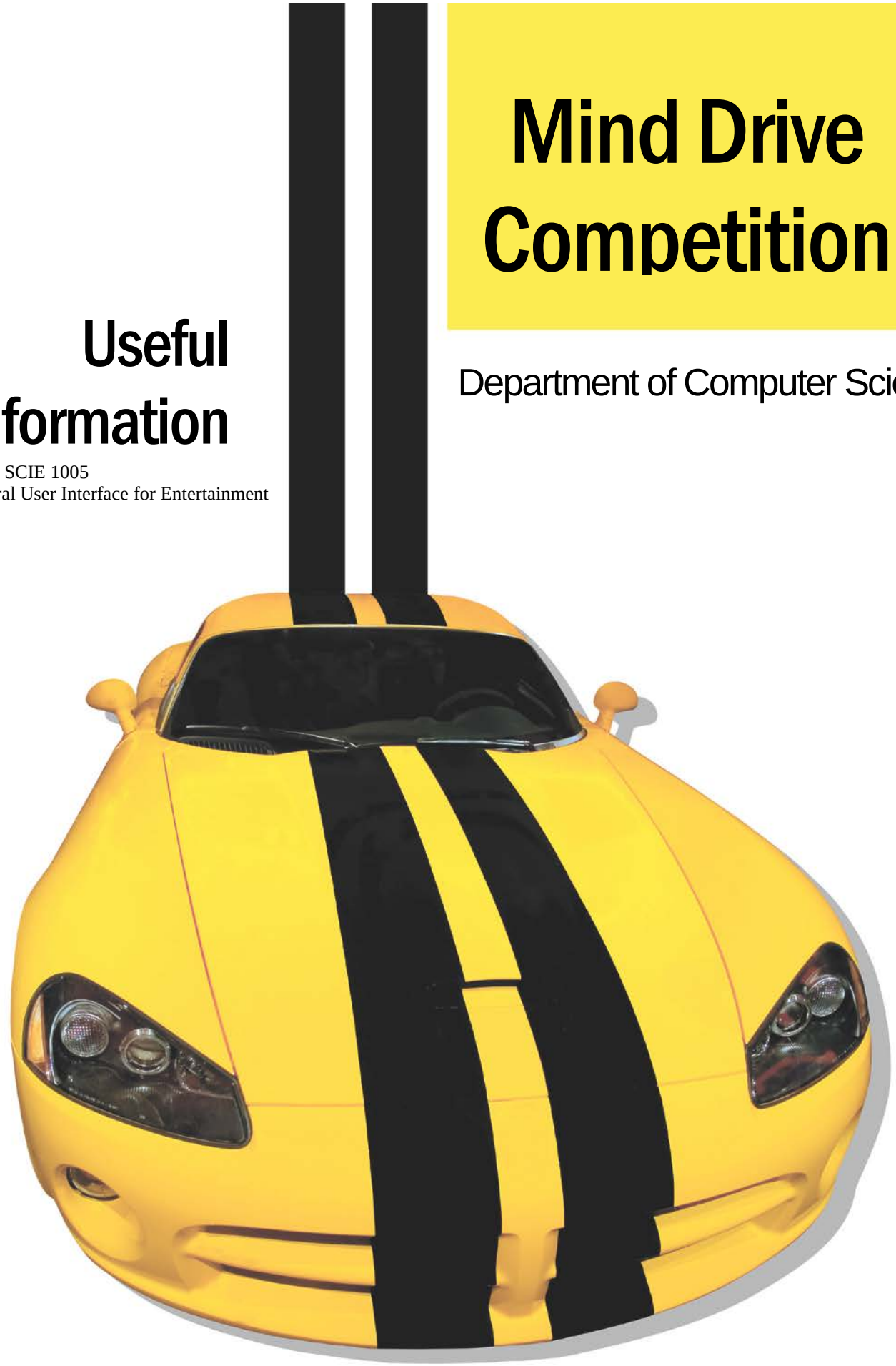
# Mind Drive Competition

## Useful Information

From SCIE 1005
Natural User Interface for Entertainment

Department of Computer Science



## Background:

1. **Natural User Interface**

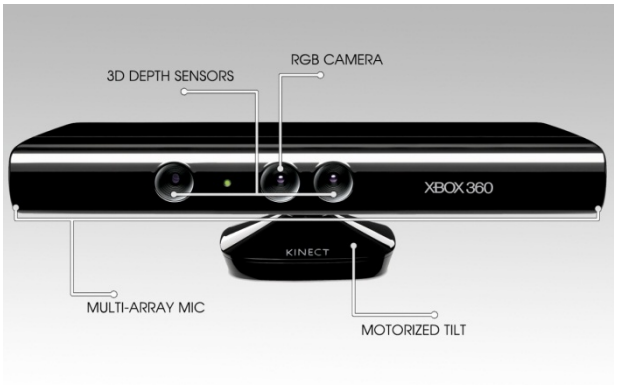



**Figure 1. A front view of a Kinect senor**

A Human Machine Interface is an interface which allows interactions between a human being and a machine. A user needs some ways to tell the machine what to do. Examples of input devices include keyboards, mice, and touch screens.

A Natural User Interface is just "natural to its user". A user may communicate through gestures, expressions, movements, voices, or any other natural means that you can think of.

*Think: What is the impact of the invention of natural user interface?*

In this lab, we study a natural user interface, called Microsoft Kinect.

Kinect is a motion sensing device, designed by Microsoft and for Xbox 360 game console, originally released in 2010. It has a RGB camera senor, a 3D depth senor and a microphone for audio input. It captures not only what the objects look like, but also where they are in space. It could capture audio and color, and simulate the 3D spatial data to generate skeleton data. For example, the images it captured contain accurate 3D informaiton (X, Y, and Z coordinates) of one or two human objects. Kinect produces such data to a computer or game console in the form of data streams. Kinect analyzes the streams and recognizes the intentional inputs to the computer.

### 1.1.Skeleton

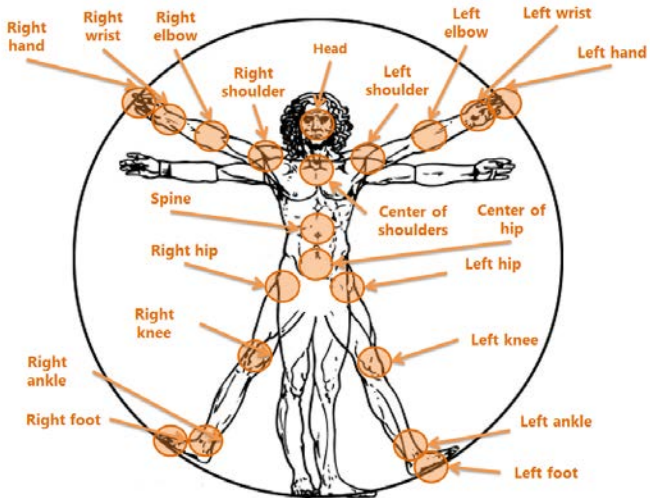



**Figure 2. Skeleton of a person defined by Kinect**

A person is detected by the Kinect sensor in skeleton form. The sensor generates an image stream and the computer extracts important joints of person(s) (see Figure 2) from the images. In default standing mode, twenty joints are continuously tracked. And, essentially, the joints of the upper part of the body are tracked in seated mode. This skeleton represents a person's current position and pose.

Each joint has a Position that reports the X, Y and Z coordinates of the joint. The coordinates are relative to the skeleton space in which zero position being at the center of the Kinect sensor. The coordinates are expressed in meters. The X axis ranges from -2.2 to 2.2 with the positive X axis extends to the left (with respect to the Kinect sensor). The Y axis ranges from -1.6 to 1.6 with the positive Y axis extends upward. The Z axis, however, is simulated and ranges from 0 to 4 with positive Z axis extends in the direction in which the Kinect sensor points.

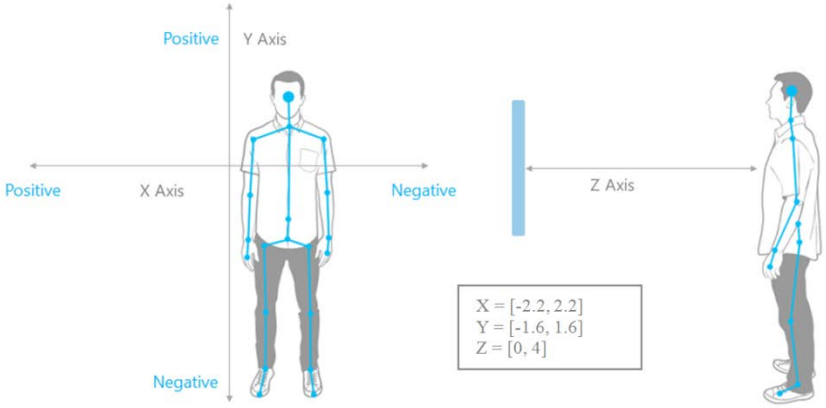



**Figure 3. Joint position defined by Kinect**

## 2. Integrated Development Environment (IDE)

One popular Computer Science related job is software developer. There are many tools to help developing their software. In particular, an integrated development environment (IDE) is a large “playground” that contains many nice facilities – code editor, builder and debugger, to help software development.

Integrated development environment often supports multiple programming languages. Developers need to be familiar with one environment to get their work going. An analogy is that when you go to playgrounds in different countries, you expect to see similar facilities in the playgrounds. The only difference is that you need to speak different langauges when interacting with people there.

There are many programming languages. Each of them is designed and best for certain purposes. For exmaple, Processing is used for electronic arts and visual design. In this lab, we use C# - one of the programming languages supported by Kinect for Windows SDK.

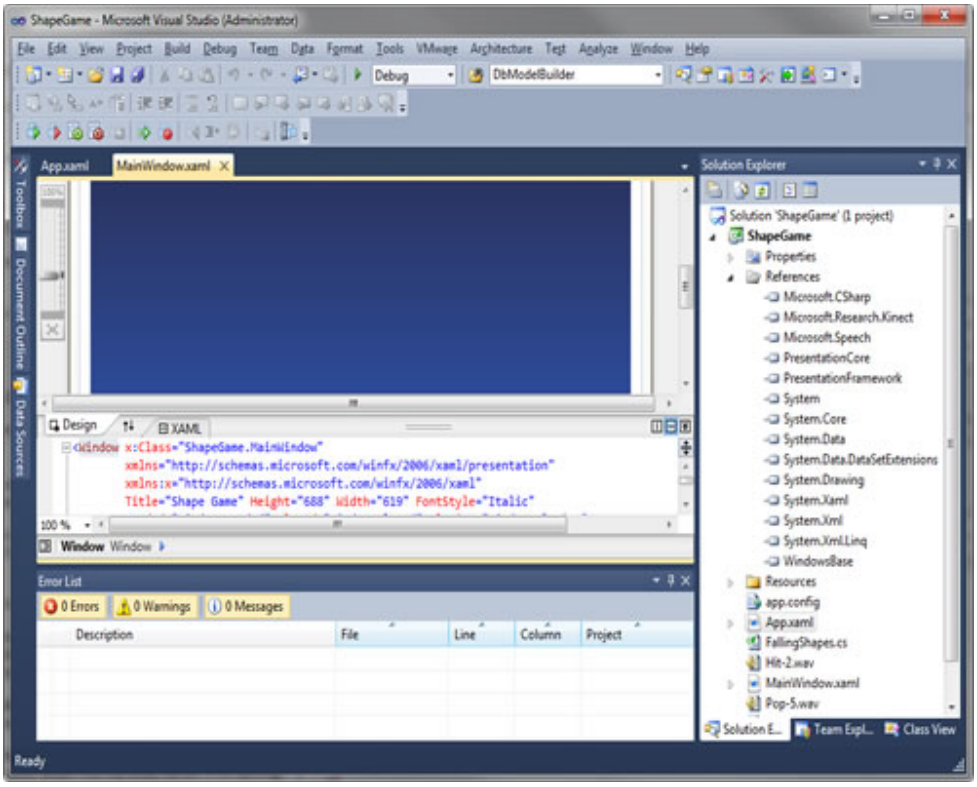



**Figure 4. An example of integrated development Environment — Visual Studio**

## 3. Source code – the file containing the program written in a programming language

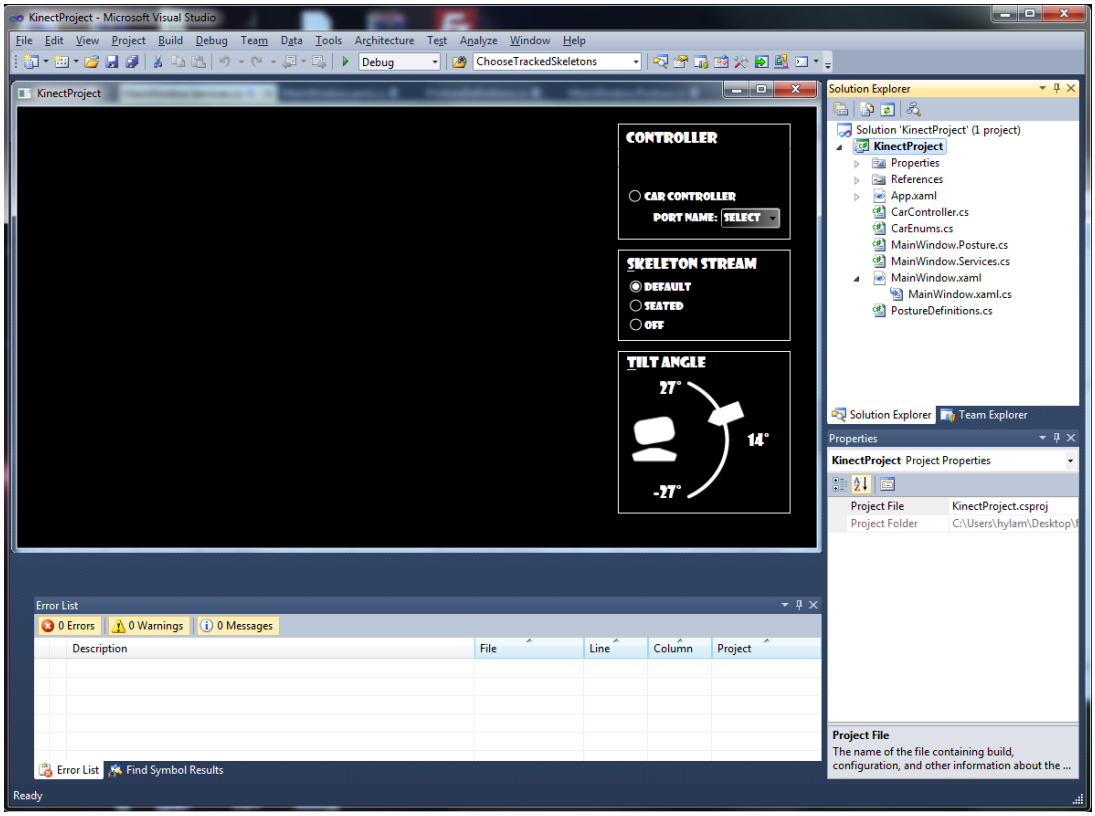



**Figure 5. Source code of this project**

The interface design of this project is in `MainWindow.xaml`. The primary purpose of the Windows in an application is to hold *display content* that enables users to interact with it. Extensible Application Markup Language (XAML) page is use for the display content in this project.

In Figure 5, the right pane shows the solution explorer. In the explorer, it shows an organized view of the code of this project. The bottom right pane shows the properties of the selected controls or items in the solution explorer.

In this competition, source codes are written in the .cs files. The overview of the source code:

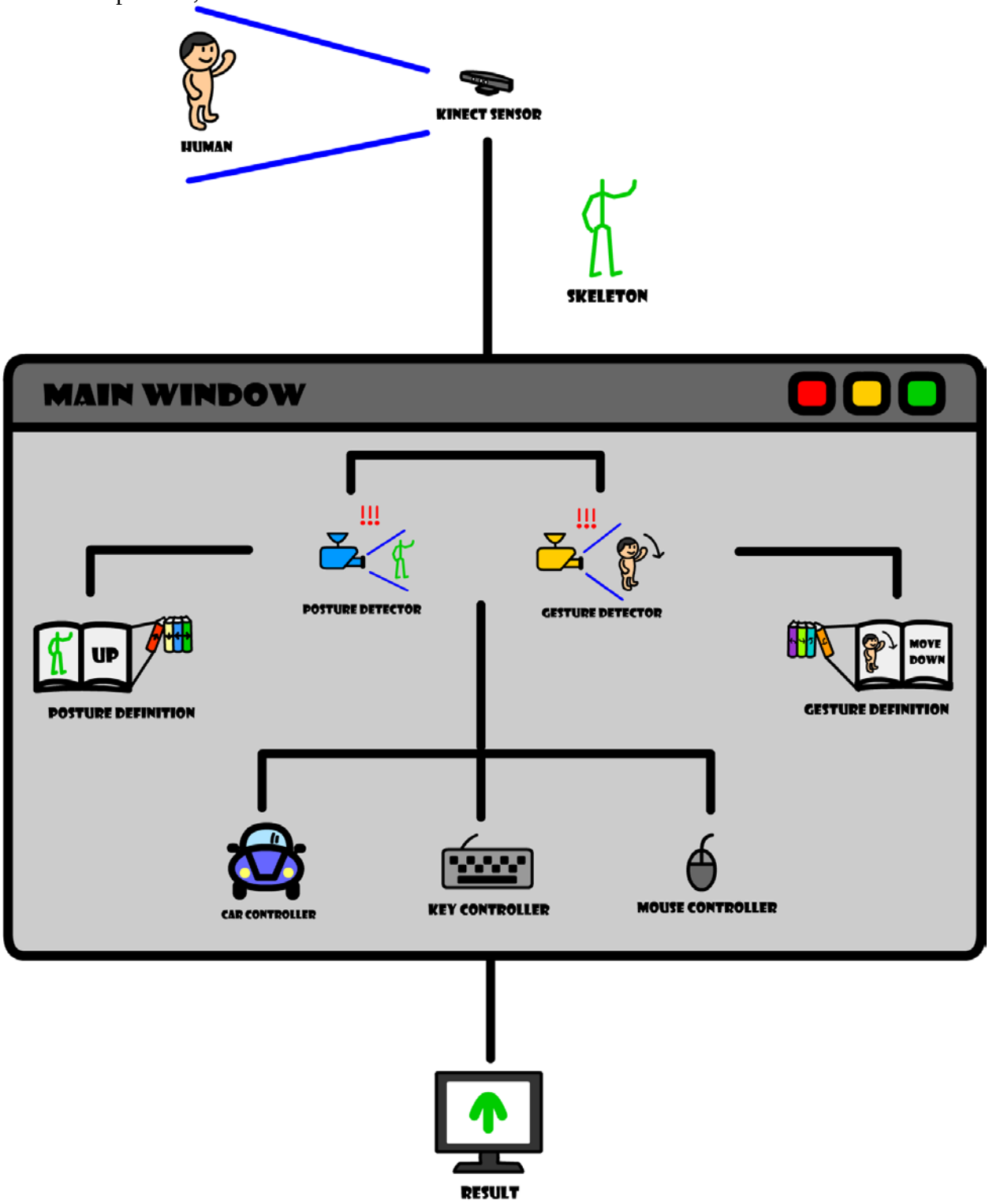


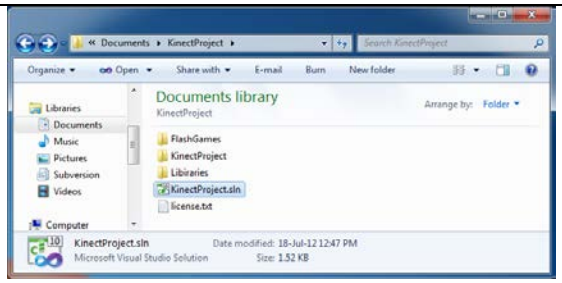

**Figure 6. The overview of the source code in the Solution Explorer**

The source code is specifying the components/objects in Figure 6. Each .cs file roughly corresponds to one object in the figure. For example, the details about *the main window* are described in `MainWindows.cs`. The description is written in C# language (surprise!)

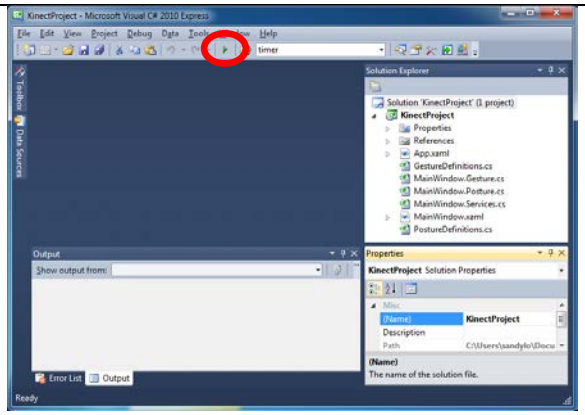
This task replaces some key commands with postures. In the following, this manual will guide you to open a project in the IDE, locate the relevant code to detect posture, compile the code and run it!

## Opening and closing a project

1. Download the resource file from Mind Drive Website.
2. Unzip the given .zip file
3. Open the Kinect Project by double-clicking the solution KinectProject.sln

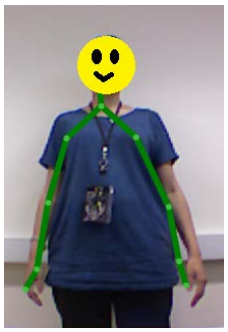



4. Click the icon or press “F5” to run the project



5. Enable the skeleton stream by choosing either DEFAULT or SEATED.
6. Adjust the tilt angle if required.


7. Show yourself in front of the Kinect camera. (You may need to make some movements to let the camera detect you.)
8. Observe the image on screen. You should see yourself tracked by a green skeleton.
9. Close the running project by clicking


SEATED


DEFAULT

Reference for **PostureDefinitions.cs**

In C#, it is written as

```
Tools.getJoint(skeleton, JointType.HandRight).Position.Y > Tools.getJoint(skeleton, JointType.Head).Position.Y
```

Explanations:

- `Tools.getJoint()` is a funciton to retrieve a joint from the skeleton data.
- The following is a list of all joints in JointType that we could get from a skeleton.

| | | | | |
|---|---|---|---|---|
| AnkleLeft | AnkleRight | ElbowLeft | ElbowRight | FootLeft |
| FootRight | HandLeft | HandRight | Head | HipCenter |
| HipLeft | HipRight | KneeLeft | KneeRight | ShoulderCenter* |
| ShoulderLeft | ShoulderRight | Spine | WristLeft | WristRight |

*Center, between shoulders

- For example, `Tools.getJoint(skeleton, JointType.HandLeft)` will give us the left hand joint
- By using Position.X, Position.Y and Position.Z, we could get the X, Y and Z coordinates from the joint respectively.
- E.g. `Tools.getJoint(skeleton, JointType.HandLeft).Position.Z` will give us the *z* coordinate of the left hand.
- " > " means "greater than". Apart from ">", we could use "<" which means "less than" to compare two coordinates.

Example:

```
if (Tools.getJoint(skeleton, JointType.HandRight).Position.Y >
    Tools.getJoint(skeleton, JointType.Head).Position.Y)
{
      if (Tools.getJoint(skeleton, JointType.HandLeft).Position.Y >
         Tools.getJoint(skeleton, JointType.Head).Position.Y)
      {
            return PostureResult.Succeed;
      }
   return PostureResult.Undetermined;
}
return PostureResult.Fail;
```

All joint relations are fulfilled

Part of joint relations is fulfilled

None of joint relations is fulfilled