CPE 658

IAVA User's Manual

IAVA Team

IAVA Team
CPE658 – Spring 2012
4/24/2012

SIGNATURES

_____ _____

Customer Signature Date

_____ _____

Instructor Signature Date

_____ _____

Developer Signature Date

_____ _____

Developer Signature Date

_____ _____

Developer Signature Date

REVISIONS

Revision Number	Date	Description
1.0	April 24, 2012	Created initial document.

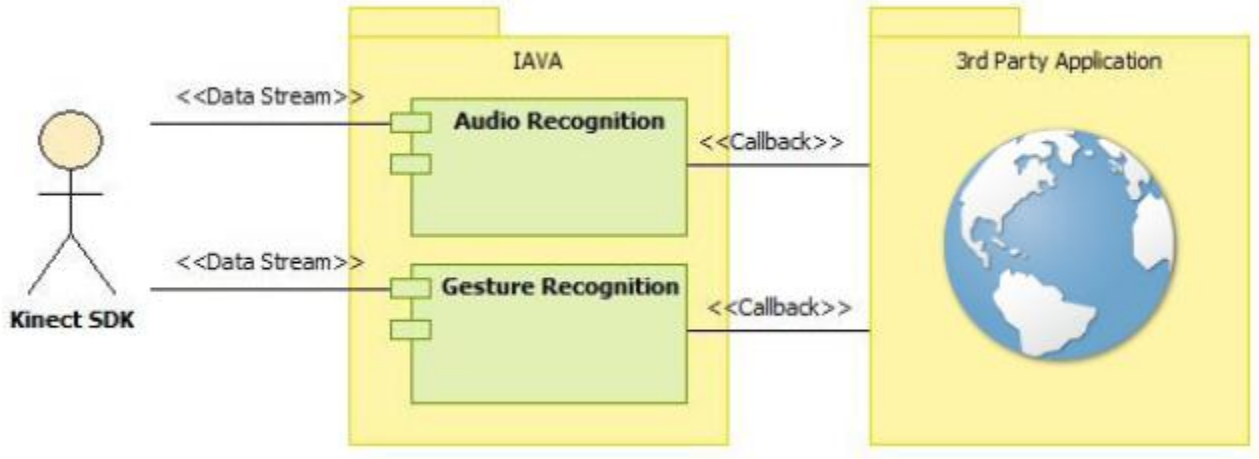
TABLE OF CONTENTS

1. INTRODUCTION

1.1. SYSTEM OVERVIEW

The Interactive Audio Visual API (IAVA) is an application programming interface that provides a simple and intuitive way to add audio and gesture recognition support to a client application. IAVA uses the Microsoft Kinect device to capture audio and gestures data. The purpose of IAVA is to abstract the complexities involved with using the Kinect Software Development Kit (SDK) to interpret gestures and the Microsoft Speech SDK to recognize speech. The client application simply calls IAVA and lets it perform the heavy lifting. IAVA is designed as a callback-based API; this means that the client application will tell IAVA what audio and gesture commands to watch for and IAVA will alert the client application when one is recognized.

1.2. SYSTEM ARCHITECTURE



IAVA sits between the Microsoft Kinect and the 3rd party application. IAVA handles the raw data streams coming from the Kinect, and processes it through the Audio Recognition and Gesture Recognition components. An application can configure these components to listen for certain audio commands and/or look for specific gestures. When the commands are detected, IAVA will trigger the callback the application registered. At this point the 3rd party application can handle the command in the appropriate manner.

2. SYSTEM REQUIREMENTS

2.1. HARDWARE REQUIREMENTS

- 32-bit (x86) or 64-bit (x64) processor
- Dual-core 2.66-GHz or faster processor
- Dedicated USB 2.0 bus
- 2 GB RAM
- A Microsoft Kinect for Windows sensor or Kinect for XBox 360 sensor

2.2. SOFTWARE REQUIREMENTS

- Microsoft® Visual Studio® 2010 Express or other Visual Studio 2010 edition
- .NET Framework 4.0
- Microsoft Kinect for Windows SDK 1.0
- Must be developing in the C# language.

3. INSTALLATION

3.1. INSTALL THE KINECT SDK

http://www.microsoft.com/en-us/kinectforwindows/develop/overview.aspx

1) Make sure the Kinect sensor is not plugged into any of the USB ports on the computer.
2) If you installed a previous version of the Kinect for Windows SDK, you must uninstall it before proceeding.
3) Remove any other drivers for the Kinect sensor.
4) Uninstall any Microsoft Speech runtime components including both the x86 and x64 bit versions, plus the Kinect Language Pack.
5) Close Visual Studio. You must close Visual Studio before installing the SDK and then restart it after installation is complete to pick up environment variables that the SDK requires.
6) From the download location, double-click the installer. This single installer works for both 32bit and 64bit Windows.
7) Once the SDK has completed installing successfully, ensure the Kinect sensor is plugged into an external power source and then plug the Kinect sensor into the PC's USB port. The drivers will load automatically.
8) The Kinect sensor should now be working correctly.

3.2. INSTALL THE IAVA API

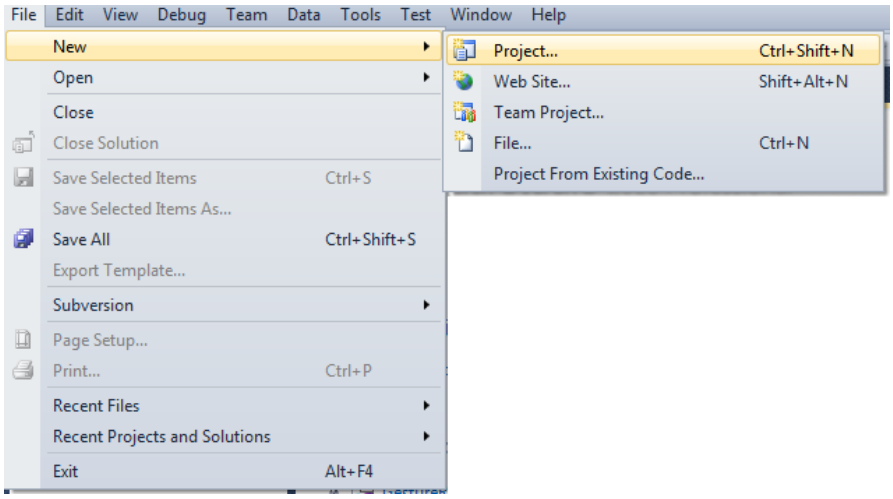
http://code.google.com/p/cpe-656-helix-kinect/downloads/detail?name=IAVA%20Installer.zip&can=2&q

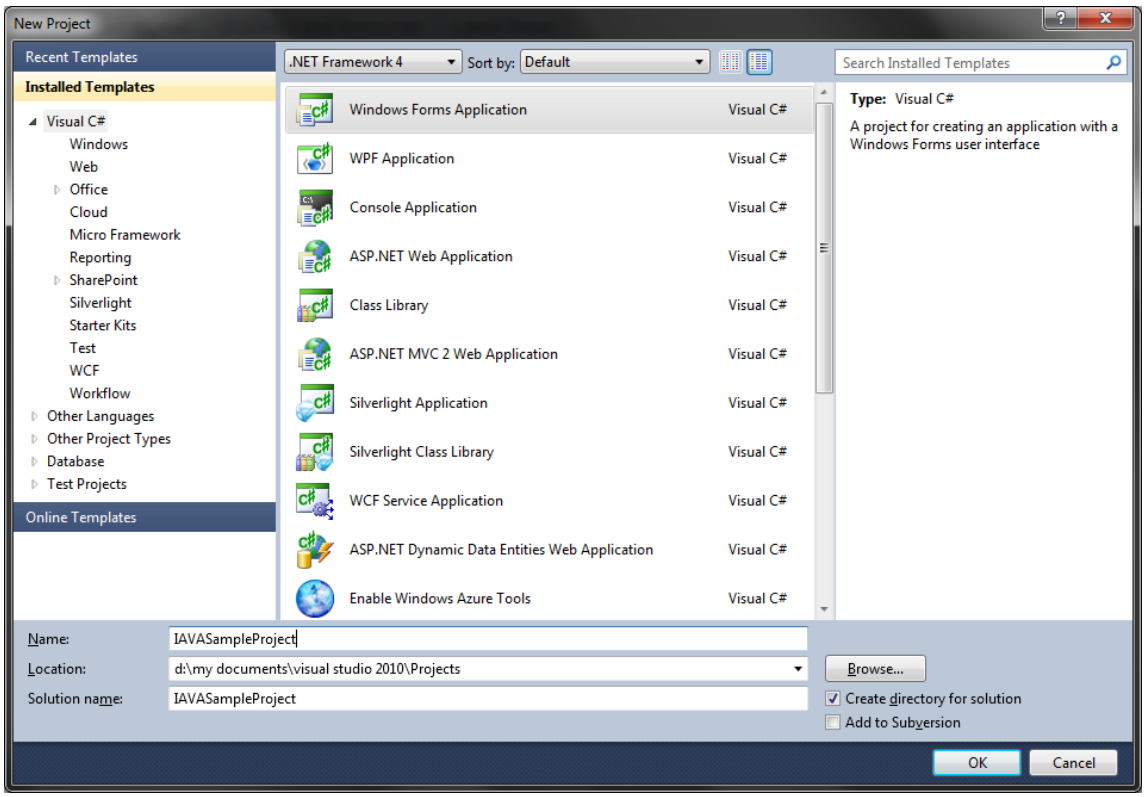
1) From the download location, double-click the installer. This single installer works for both 32bit and 64bit Windows.
2) Unzip the files to a location of your choice. These DLL files will be referenced by the project you are creating. A good idea would be to copy the DLL's into a folder located in your project.

4. USING IAVA

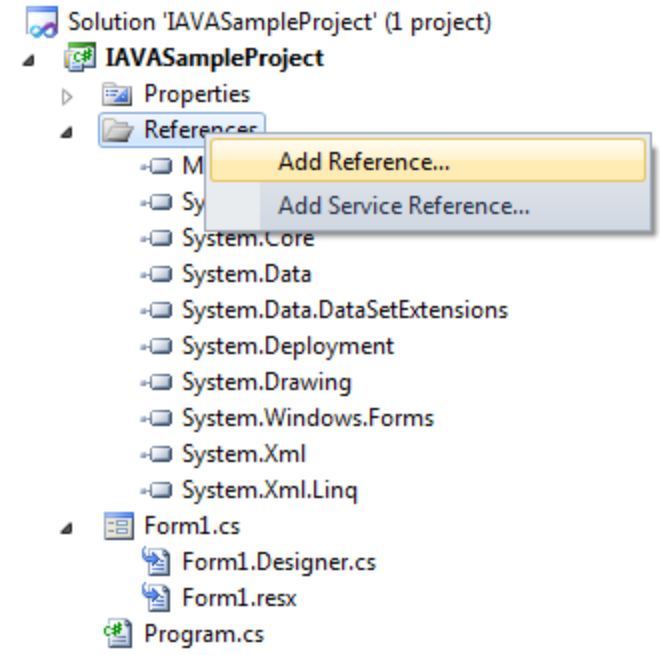
4.1. REFERENCING THE IAVA API

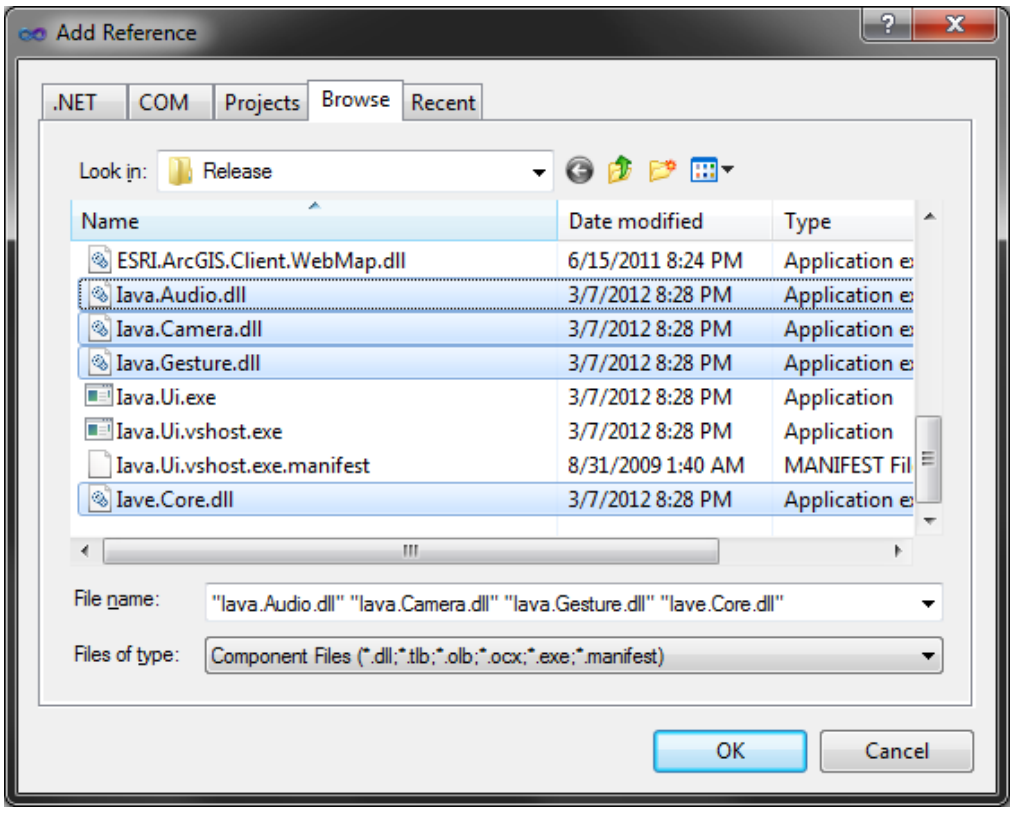
1) Create a new project

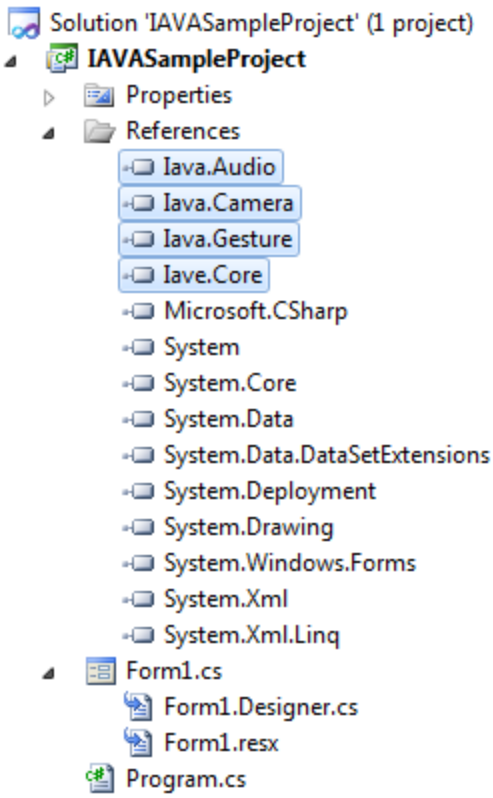




2) Add the reference in your project to the DLLs that you need in your application. This will vary depending on whether you need gesture or audio recognition.


Solution 'IAVASampleProject' (1 project)
IAVASampleProject
Properties
References
Add Reference...
Add Service Reference...
System.Core
System.Data
System.Data.DataSetExtensions
System.Deployment
System.Drawing
System.Windows.Forms
System.Xml
System.Xml.Linq
Form1.cs
Form1.Designer.cs
Form1.resx
Program.cs


Add Reference
.NET
COM
Projects
Browse
Recent
Look in:
Release
Name
Date modified
Type
ESRI.ArcGIS.Client.WebMap.dll
6/15/2011 8:24 PM
Iava.Audio.dll
3/7/2012 8:28 PM
Iava.Camera.dll
3/7/2012 8:28 PM
Iava.Gesture.dll
3/7/2012 8:28 PM
Iava.Ui.exe
3/7/2012 8:28 PM
Application
Iava.Ui.vshost.exe
3/7/2012 8:28 PM
Application
Iava.Ui.vshost.exe.manifest
8/31/2009 1:40 AM
MANIFEST Fil
Iave.Core.dll
3/7/2012 8:28 PM
File name:
"Iava.Audio.dll" "Iava.Camera.dll" "Iava.Gesture.dll" "Iave.Core.dll"
Files of type:
Component Files (*.dll;*.tlb;*.olb;*.ocx;*.exe;*.manifest)
OK
Cancel

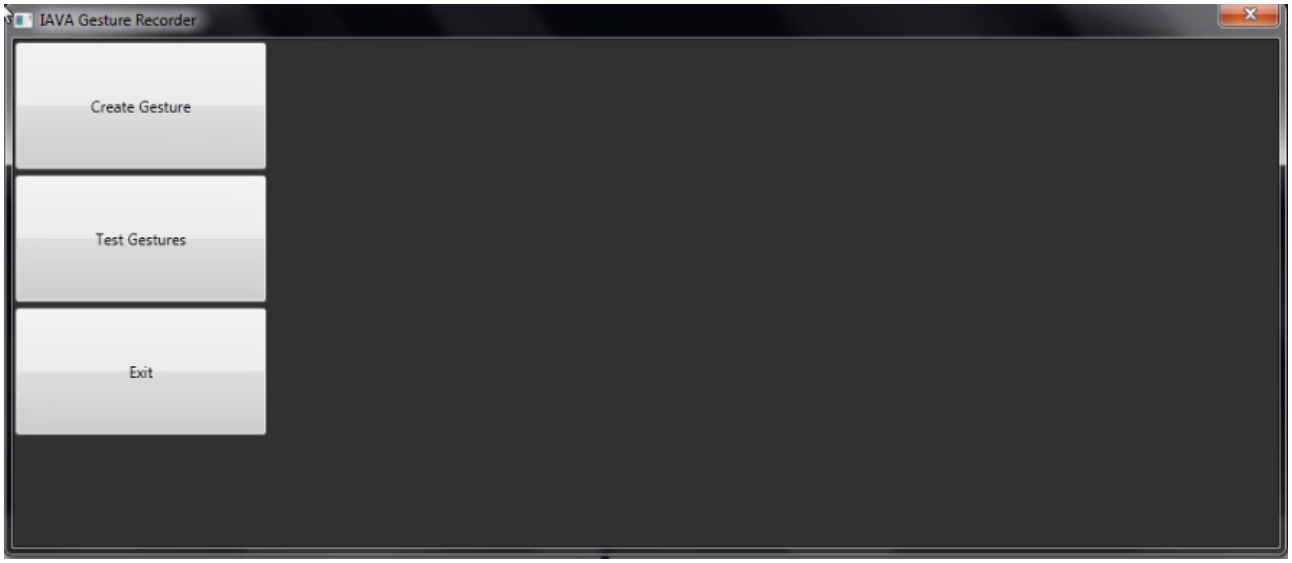


You are now ready to use the IAVA API.

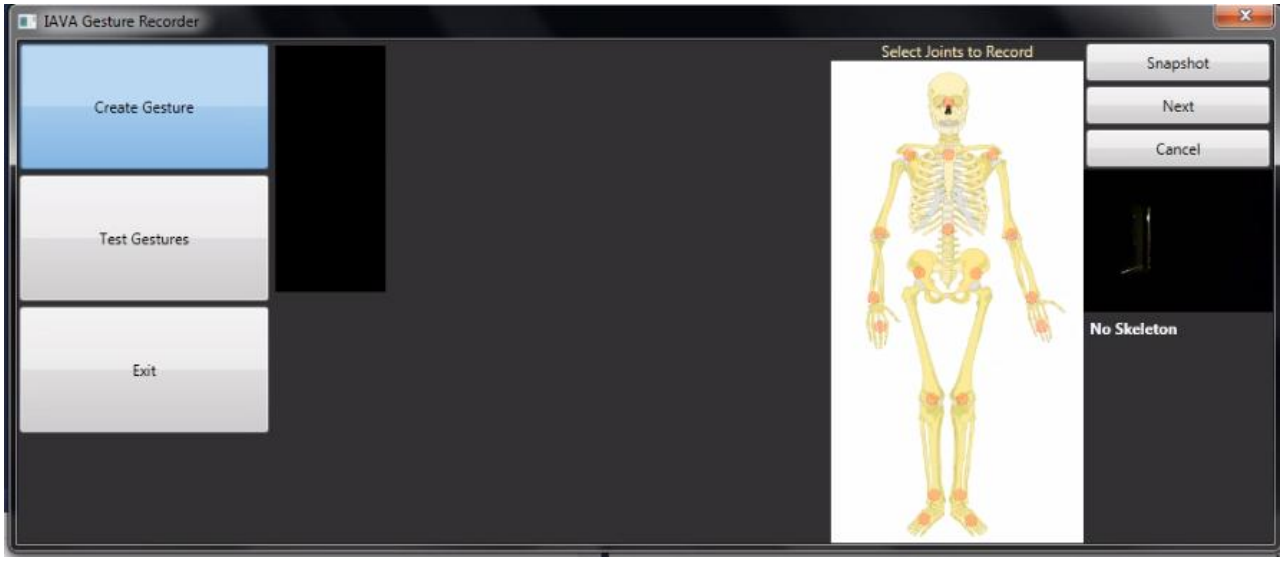
DLLName	Description & Use
Iava.Core	This is the main IAVA dll. If you wish to use any of the IAVA functionality, you will need to reference this library.
Iava.Audio	This is the dll containing everything needed for audio recognition. If you wish to use the IAVA audio recognition functionality, you will need to reference this library.
Iava.Input	This is the dll containing everything needed to utilize the Kinect device directly. If you wish to access the Kinect cameras, you will need to reference this library.
Iava.Gesture	This is the dll containing everything needed for gesture recognition. If you wish to use the IAVA gesture recognition functionality, you will need to reference this library.

4.2. CREATING A GESTURE

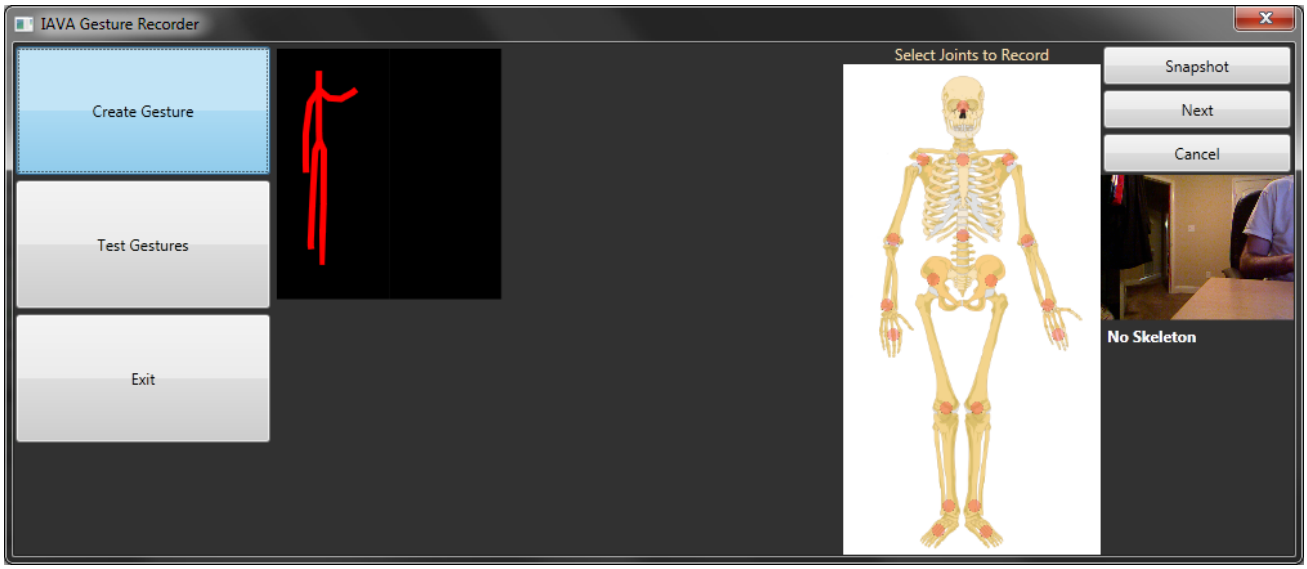
1) Open the GestureRecorder Application



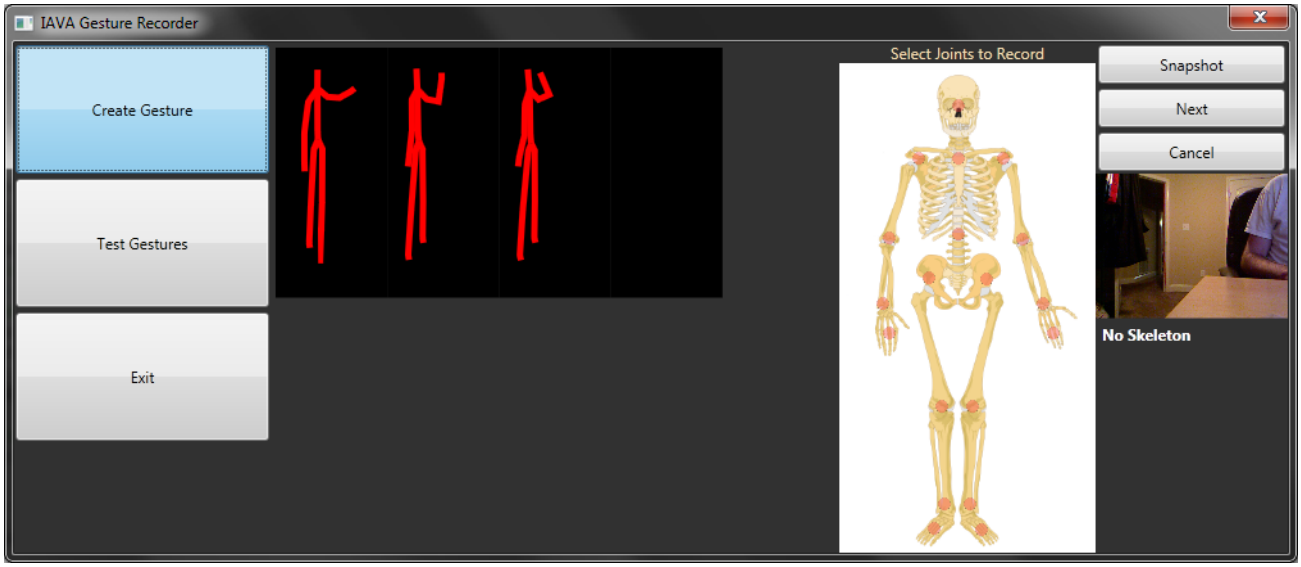
2) Click the **Create Gesture** button



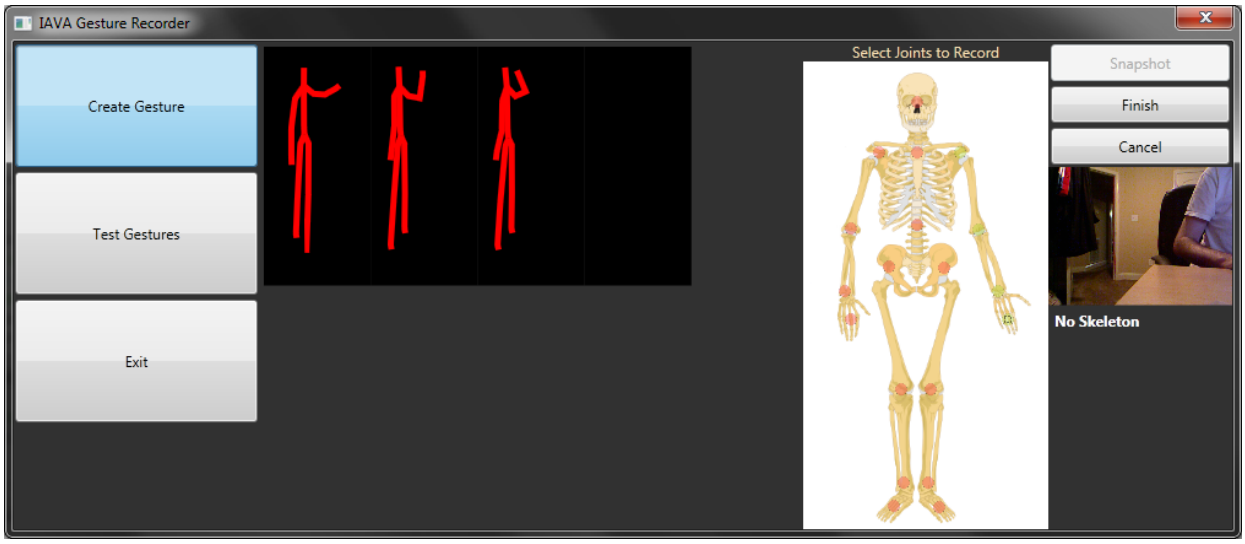
3) Each snapshot is a sequential state of the gesture to look for. Each snapshot has to be detected in order for the entire gesture to be recognized. In order to take a snapshot, you can either click the **Snapshot** button or say '***IAVA Snapshot'***.



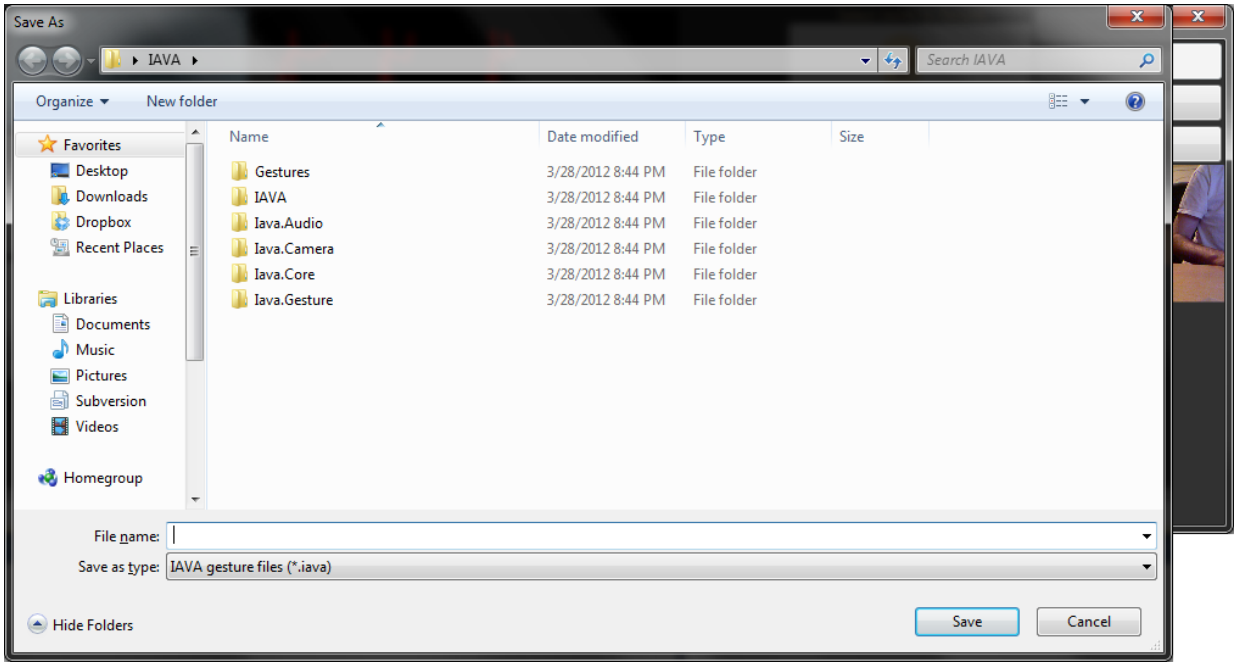
4) Repeat this for each segment of your gesture.



5) After you have finished recording the snapshots, click **Next** button or say '***Next***'.
6) Now select the joint(s) of the body that you are interested in tracking for the gesture. For example, in a Wave gesture you are only interested in watching the arm joints.



7) After you have selected your joints of interests, click **Finish** to save your gesture to a file.



8) Click Ok after you have found a location to save your gesture file.

5. SAMPLE CODE

5.1. AUDIO RECOGNITION

The steps to detecting audio commands thru IAVA are as follows:

1) Create an **AudioRecognizer** object.

2) Subscribe to a keyword that you are trying to detect.
3) Create your callback.
4) Start the **AudioRecognizer**.

```
Iava.Audio.AudioRecognizer audioRecognizer = new Iava.Audio.AudioRecognizer();

public Form1()
{
    InitializeComponent();

    // Subscribe to the audio command we are trying to detect
    audioRecognizer.Subscribe("MyCommand", MyCommandCallback);

    // Start the AudioRecognizer
    audioRecognizer.Start();
}

private void MyCommandCallback(Iava.Audio.AudioEventArgs e)
{
    // Handle the audio command.
}
```

5.2. GESTURE RECOGNITION

The steps to detecting audio commands thru IAVA are as follows:

1) Create a **GestureRecognizer** object.
2) Subscribe to a gesture that you are trying to detect.
3) Create your callback.
4) Start the **GestureRecognizer**.

```
Iava.Gesture.GestureRecognizer gestureRecognizer =
    new Iava.Gesture.GestureRecognizer(@"C:\Gesture Folder");

public Form1()
{
    InitializeComponent();

    // Subscribe to the gesture command we are trying to detect
    gestureRecognizer.Subscribe("MyCustomGesture", MyCustomGestureCallback);

    // Start the GestureRecognizer
    gestureRecognizer.Start();
}

private void MyCustomGestureCallback(Iava.Gesture.GestureEventArgs e)
{
    // Handle the gesture command
}
```

5.3. RECOGNIZER STATUS CHANGED

1) Create an **AudioRecognizer** or **GestureRecognizer** object.
2) Register with the ***StatusChanged*** event.

```
Iava.Audio.AudioRecognizer audioRecognizer = new Iava.Audio.AudioRecognizer();

public Form1()
{
    InitializeComponent();

    // Register with the AudioRecognizer StatusChanged event
    audioRecognizer.StatusChanged += OnAudioRecognizerStatusChanged;
}

private void OnAudioRecognizerStatusChanged(object sender, EventArgs e)
{
    // Status changed on the AudioRecognizer
    Iava.Core.RecognizerStatus status = audioRecognizer.Status;
}
```

6. CODE REPOSITORY

All code for the project can be found at our IAVA Google Code repository site. You can find issues, bugs, task, and other project details at this site.

7. KNOWN ISSUES

- The gesture recognizer does not return a velocity on how fast the gesture is performed.
- The audio recognizer has issues in large rooms picking up and deciphering words.
- The gesture recorder can only record one gesture at a time. After recording one gesture, it has to be closed and restarted.

8. ACRONYMS

- **AMRDEC** – Aviation Missile Research and Development Engineering Center
- **API** – Application Programming Interface
- **IAVA** – Interactive Audio Visual API
- **ROM** – Rough Order of Magnitude
- **SDD** – Software Design Document
- **SDK** – Software Development Kit
- **SED** – Software Engineering Directorate
- **UAH** – University of Alabama Huntsville