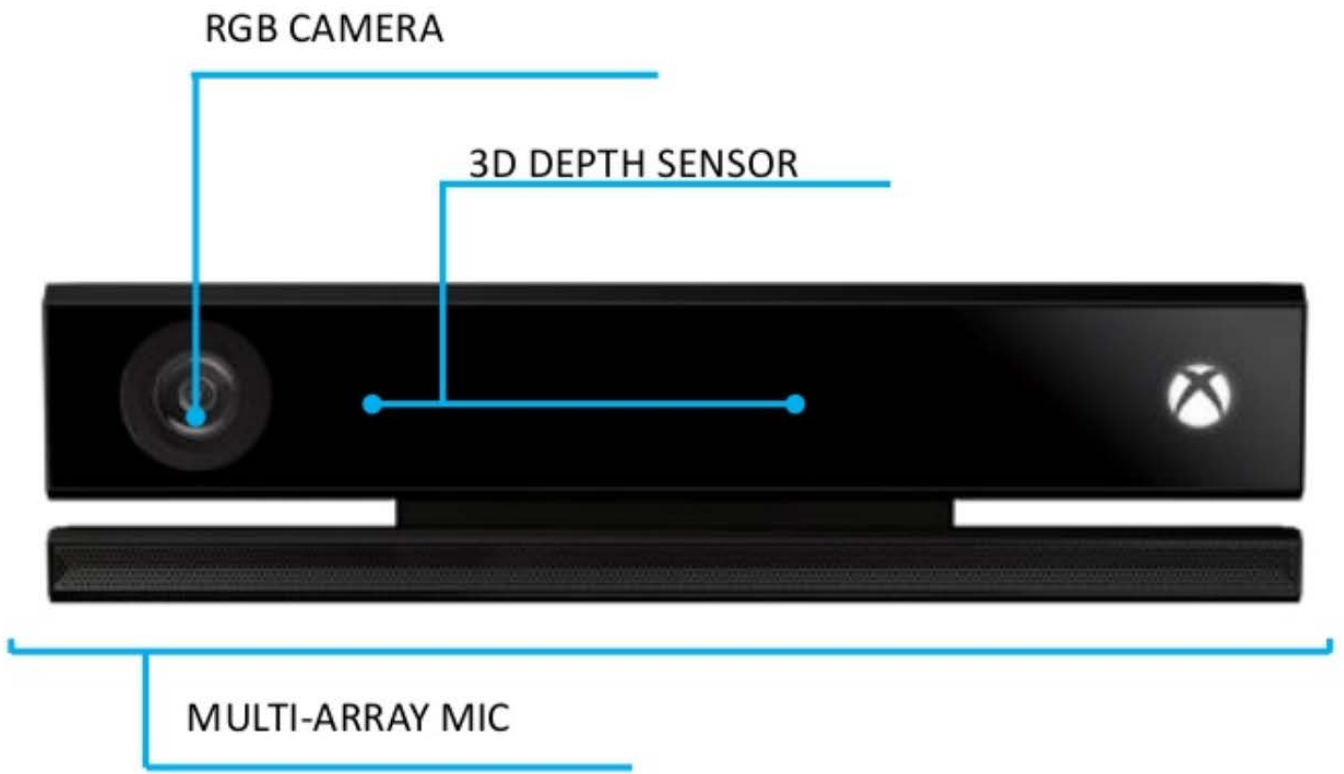
Kinect

Art with Kinect

- Puppet Parade
- Paik Times Five
- unnamed soundsculpture
- Ego
- Transcranial

Kinect 2 - Specs



Hardware:

Depth resolution:
512×424

RGB resolution:
1920×1080 (16:9)

FrameRate:
60 FPS

Latency:
60 ms

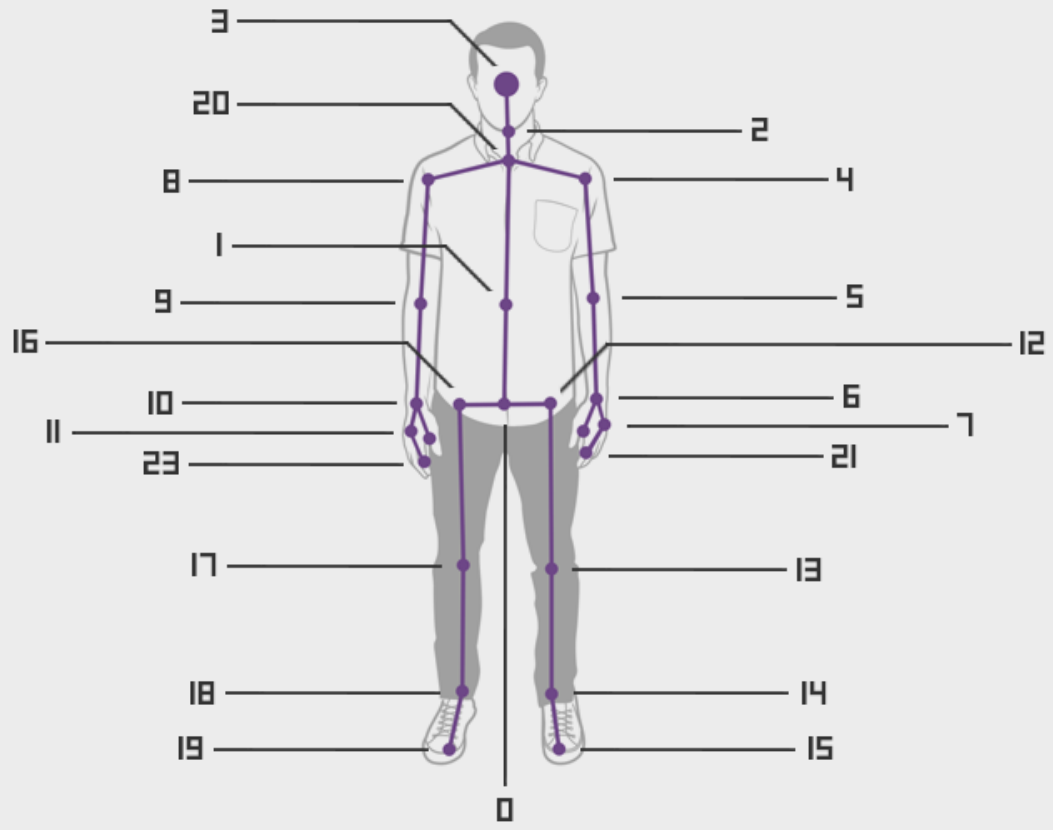
https://www.slideshare.net/MatteoValoriani/programming-with-kinect-v2

Body Tracking

KINECT V2 JOINT ID MAP

25 JOINTS
6 BODIES

JOINTTYPE_SPINEBASE	= 0.
JOINTTYPE_SPINEMID	= 1.
JOINTTYPE_NECK	= 2.
JOINTTYPE_HEAD	= 3.
JOINTTYPE_SHOULDERLEFT	= 4.
JOINTTYPE_ELBOWLEFT	= 5.
JOINTTYPE_WRISTLEFT	= 6.
JOINTTYPE_HANDLEFT	= 7.
JOINTTYPE_SHOULDERRIGHT	= 8.
JOINTTYPE_ELBOWRIGHT	= 9.
JOINTTYPE_WRISTRIGHT	= 10.
JOINTTYPE_HANDRIGHT	= 11.
JOINTTYPE_HIPLEFT	= 12.
JOINTTYPE_KNEELEFT	= 13.
JOINTTYPE_ANKLELEFT	= 14.
JOINTTYPE_FOOTLEFT	= 15.
JOINTTYPE_HIPRIGHT	= 16.
JOINTTYPE_KNEERIGHT	= 17.
JOINTTYPE_ANKLERIGHT	= 18.
JOINTTYPE_FOOTRIGHT	= 19.
JOINTTYPE_SPINESHOULDER	= 20.
JOINTTYPE_HANDTIPLEFT	= 21.
JOINTTYPE_THUMBLEFT	= 22.
JOINTTYPE_HANDTIPRIGHT	= 23.
JOINTTYPE_THUMBRIGHT	= 24.



Setup

- Have a Windows machine with USB3 port
  - Mac users -> Boot Camp
- Download and install the Kinect SDK
- Plug Kinect into USB3 port
- Test out the SDK Browser
- Download “Unity Pro Packages” from here
- Celebrate and dance for your Kinect

Documentation

- Window's documentation isn't beginner friendly...
- Programming Guide
- Main documentation page
- Some tutorials (1, 2, 3)
- Kinect SDK comes in a couple flavors – you want to find C# tutorials. They won't all be Unity-specific, but they will still be applicable.

Paid Assets

- What we saw so far is just using the “free” Windows SDK
- Kinect v2 Examples with MS-SDK ($25)
  - Wrapper around the SDK that handles common tasks, like mapping skeleton to a rig
  - Blog post – claims its free for education if you contact the developer
- Unity Mocap assets: Body, Face

Data Structures

Data Structures

- Array
  - What we've used so far
  - Fixed size, fast
  - Unity video tutorial
- List (aka Generic List)
  - Arrays that you can add to and remove from *dynamically*
  - Unity video tutorial
- Dictionary (aka Generic Dictionary)
  - Store information using a "key" (an associative array)
  - Dynamic
  - Unity video tutorial
- Wiki: Choosing the Right Data Type
- Blog: Arrays, Hashtables and Dictionaries Explained

Lists

```
List<Vector3> points = new List<Vector3>(); // Empty list

points.Add(new Vector3(10, 0, 0));          // Adding an element
points.Add(new Vector3(0, 4, 0));           // Adding another element
points.Add(new Vector3(0, 0, 2));           // Adding another element

Vector3 firstPoint = points[0];             // Accessing the 0th element
points.RemoveAt(0);                         // Removing the 0th element
int numPoints = points.Count;               // Get the number of elements

Vector3[] pointArray = points.ToArray();    // Converting list to array

points.Clear();                             // Emptying a list
```