Microsoft® .NET™ | Gadgeteer

A new way to make devices

Nicolas Villar

Sensors and Devices Research Group
Microsoft Research

Cambridge, UK

.NET Gadgeteer is a new toolkit for quickly constructing, programming and shaping new small computing devices (*gadgets*)

“From idea to working device quickly and easily”

Low threshold

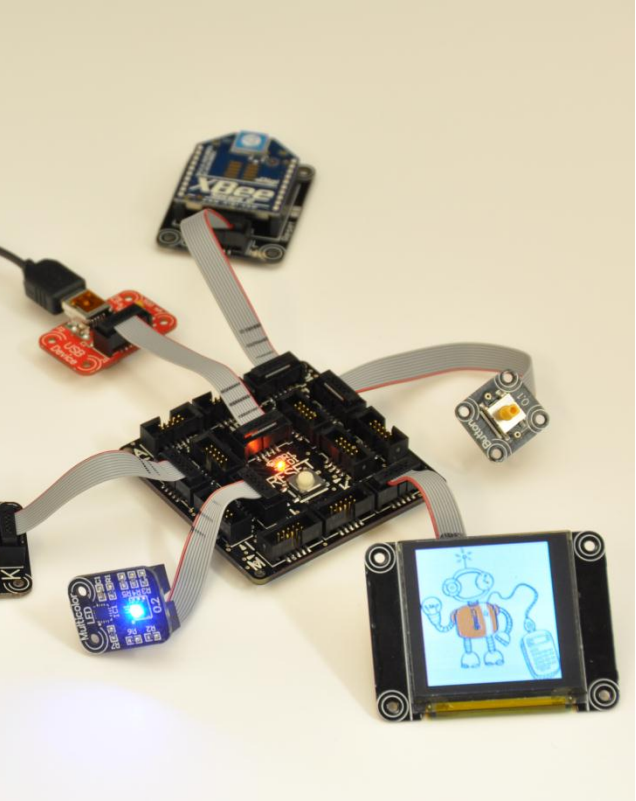
Simple gadgets should be very simple to build

High ceiling

It should also be possible to build sophisticated and complex devices

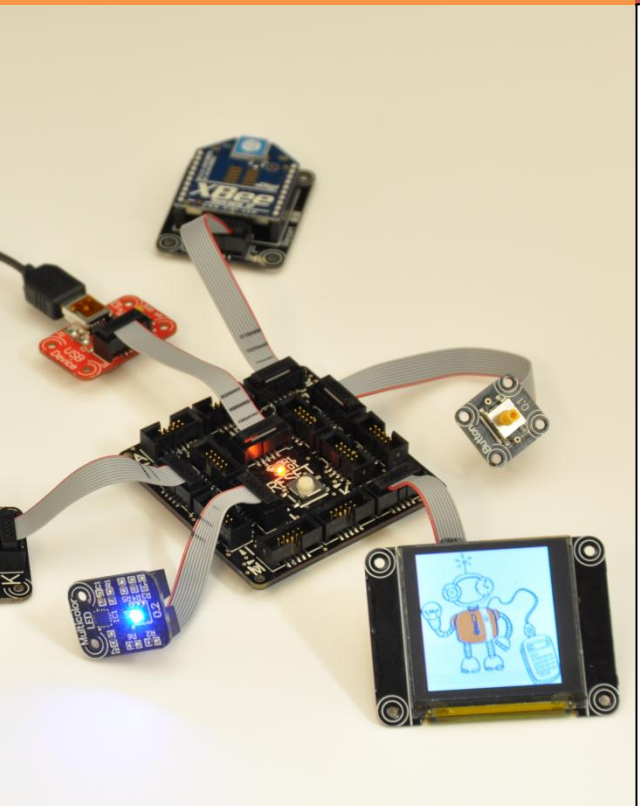
3 Key Components

Modular Hardware



3 Key Components

Modular Hardware

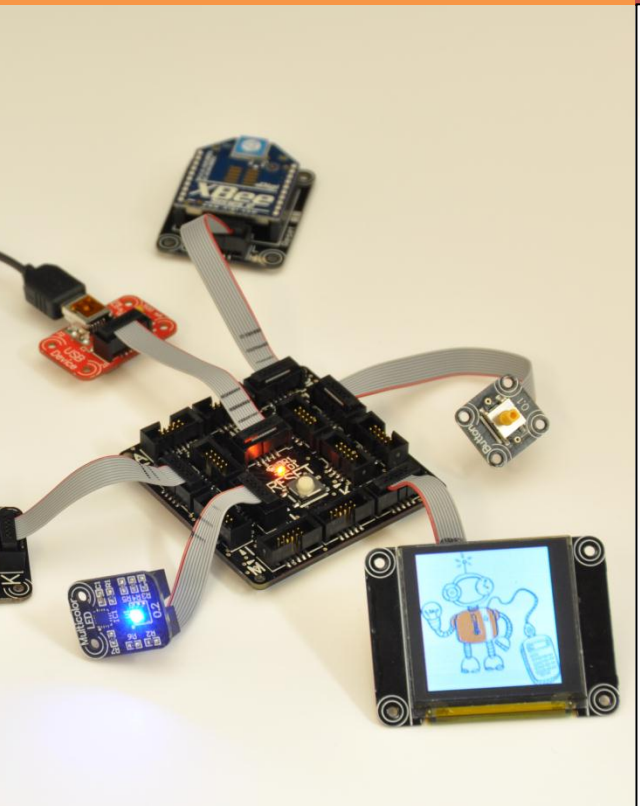
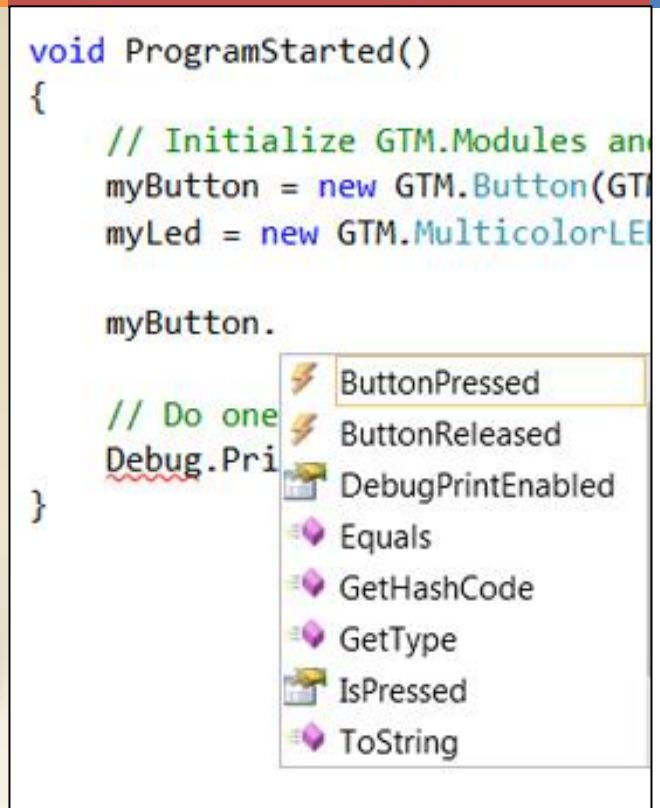
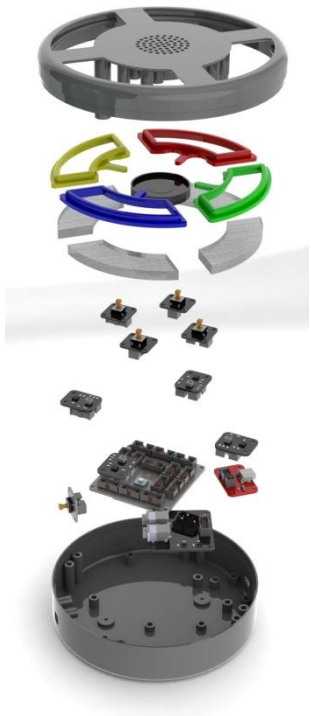


Object-Oriented Programming

```
void ProgramStarted()
{
    // Initialize GTM.Modules an
    myButton = new GTM.Button(GT
    myLed = new GTM.MulticolorLE

    myButton.

    // Do one
    Debug.Pri
}
```

- ButtonPressed
- ButtonReleased
- DebugPrintEnabled
- Equals
- GetHashCode
- GetType
- IsPressed
- ToString

3 Key Components

Modular Hardware	Object-Oriented Programming	Digital Design and Fabrication
		

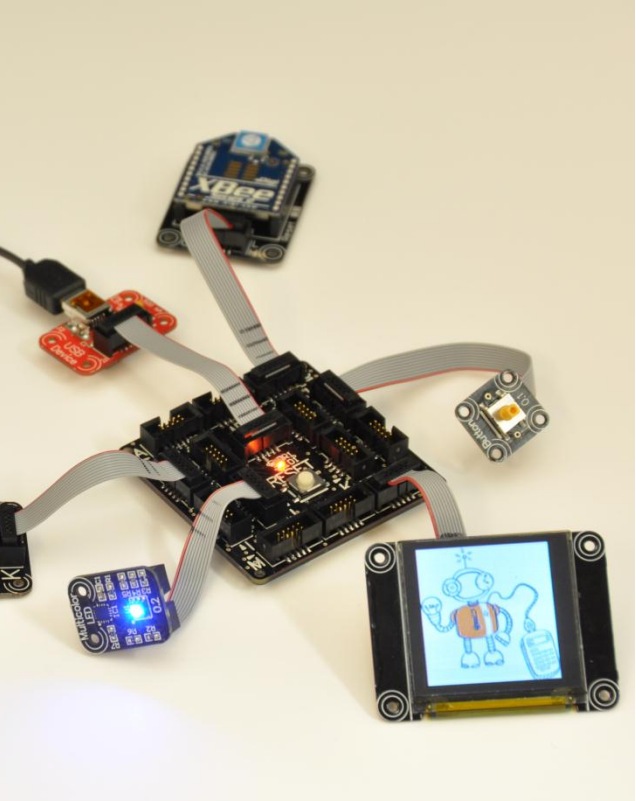
Some History

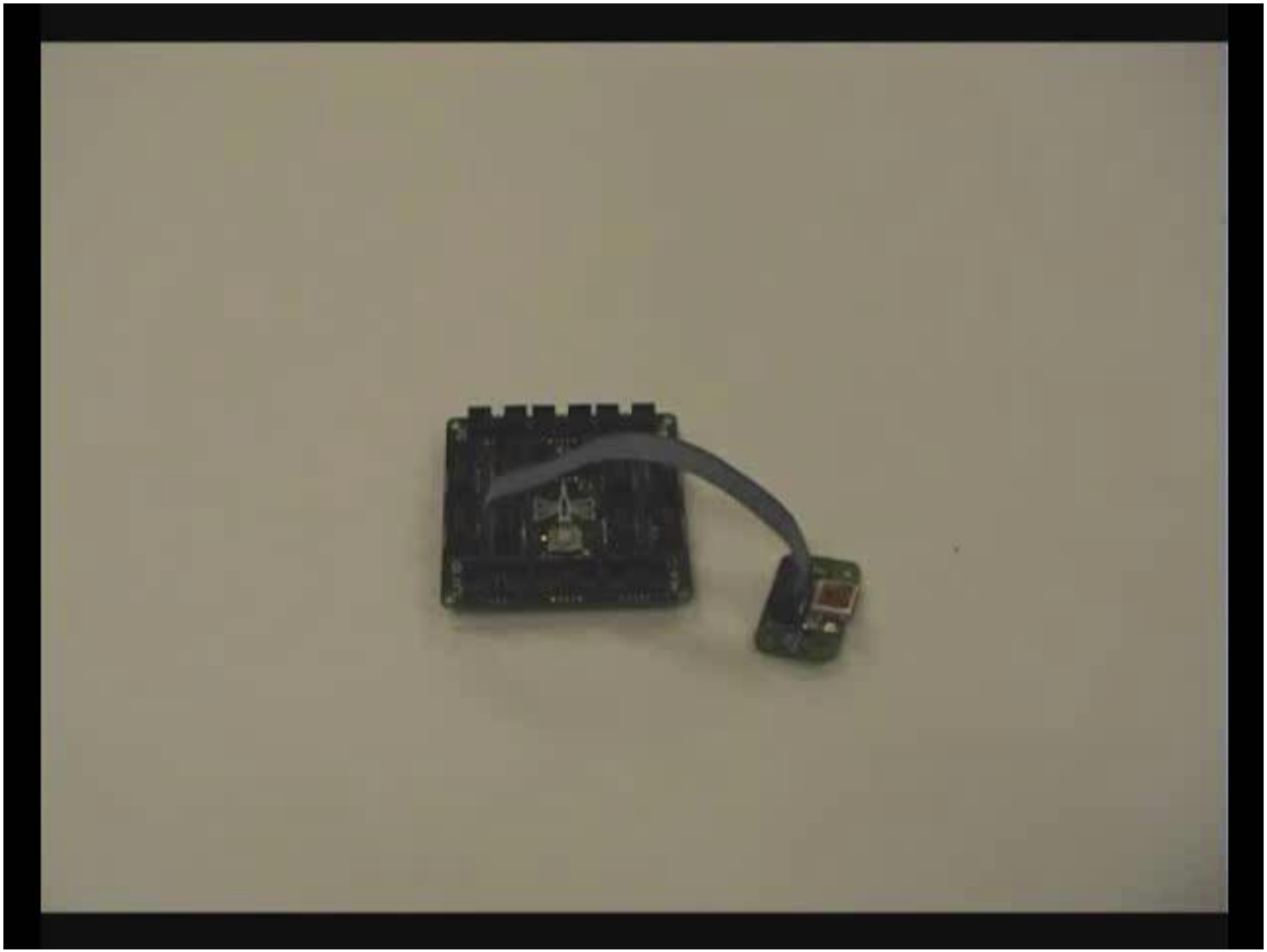
- We originally built Gadgeteer as a tool for ourselves (in Microsoft Research) to make it faster and easier to prototype new kinds of devices

- Since then, it has proven to be of interest to other researchers – but also hobbyists and educators

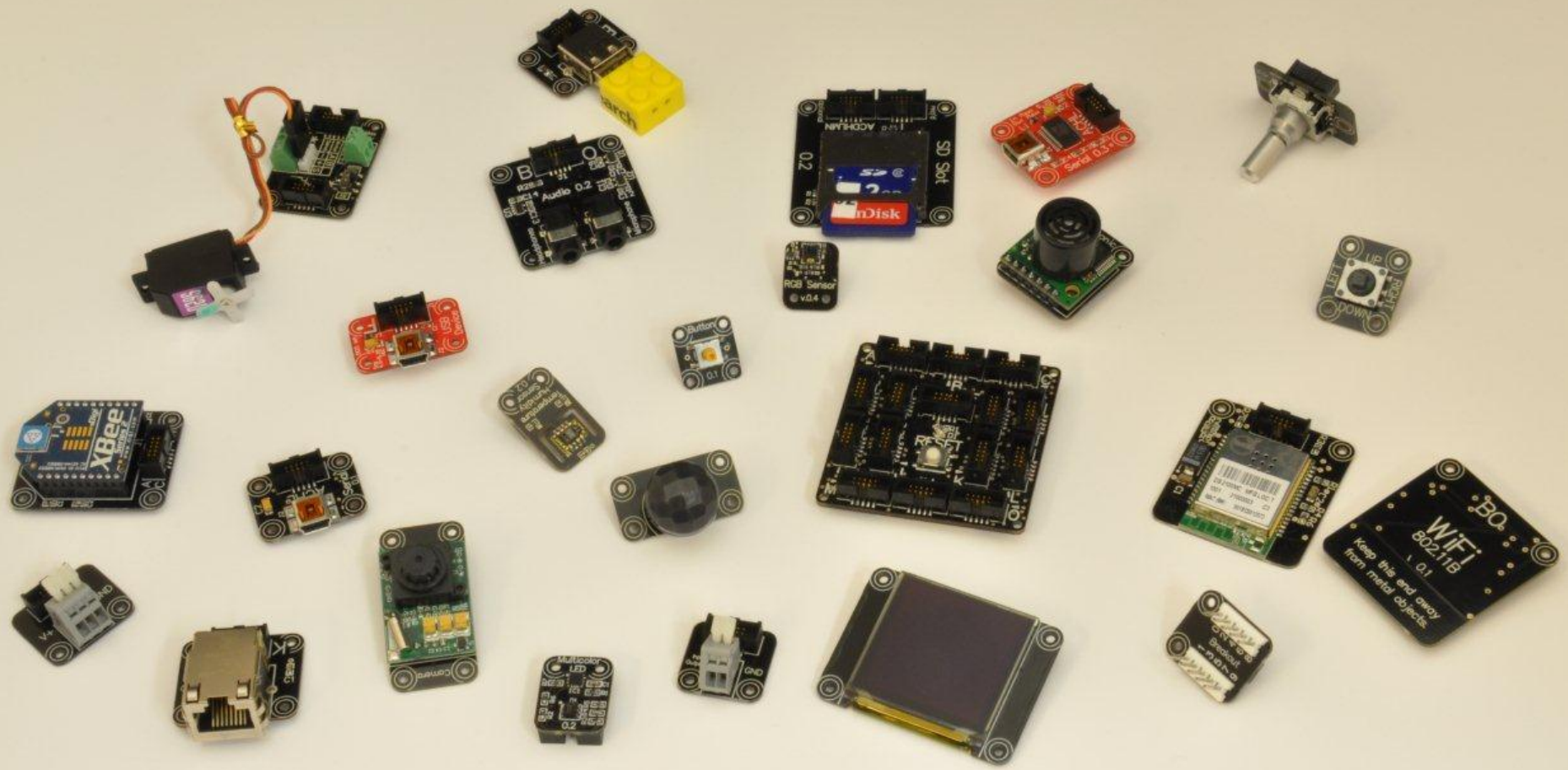
- With the help of colleagues from all across Microsoft, we are working on getting Gadgeteer out of the lab and into the hands of others

Modular Hardware

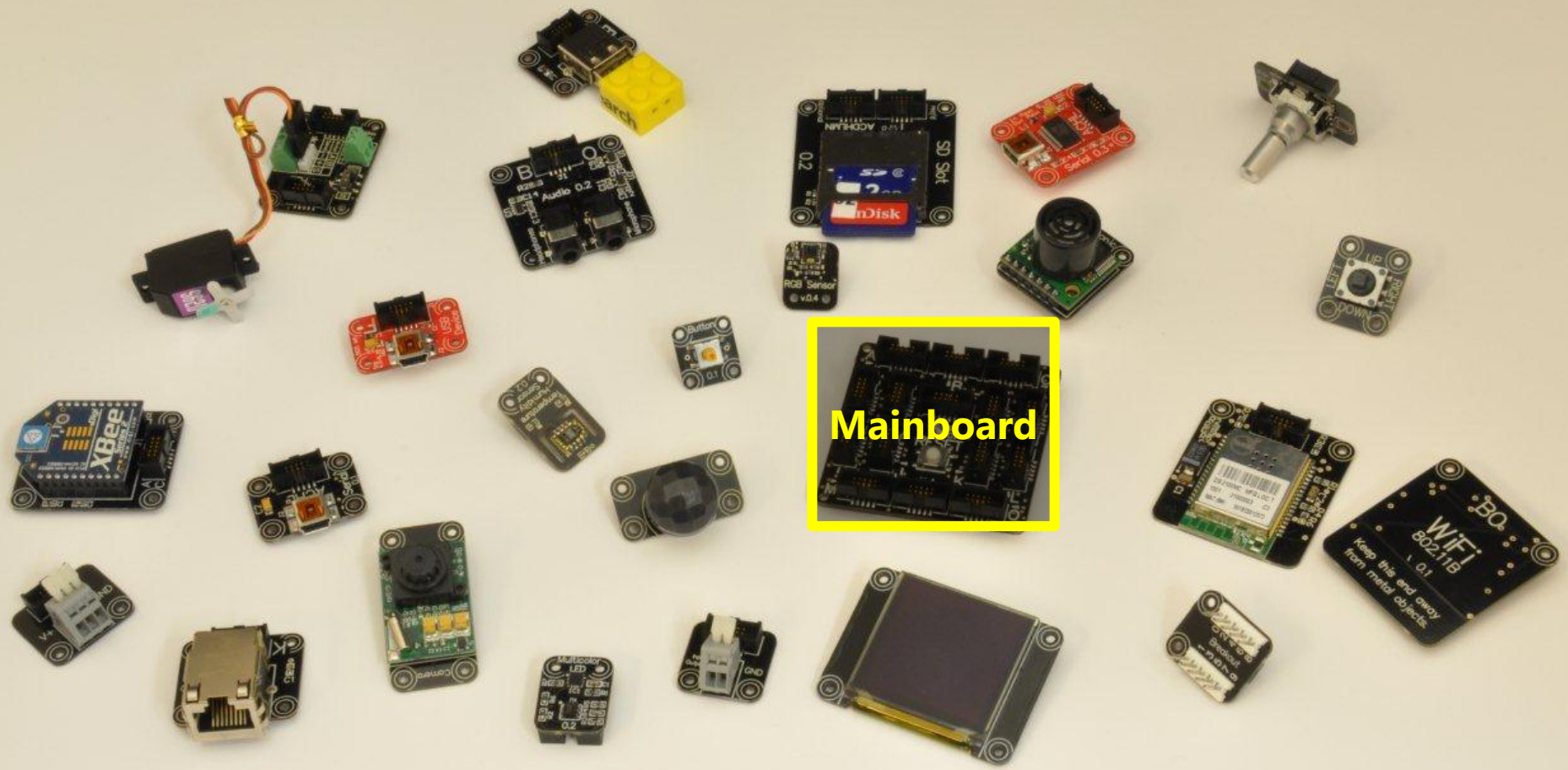




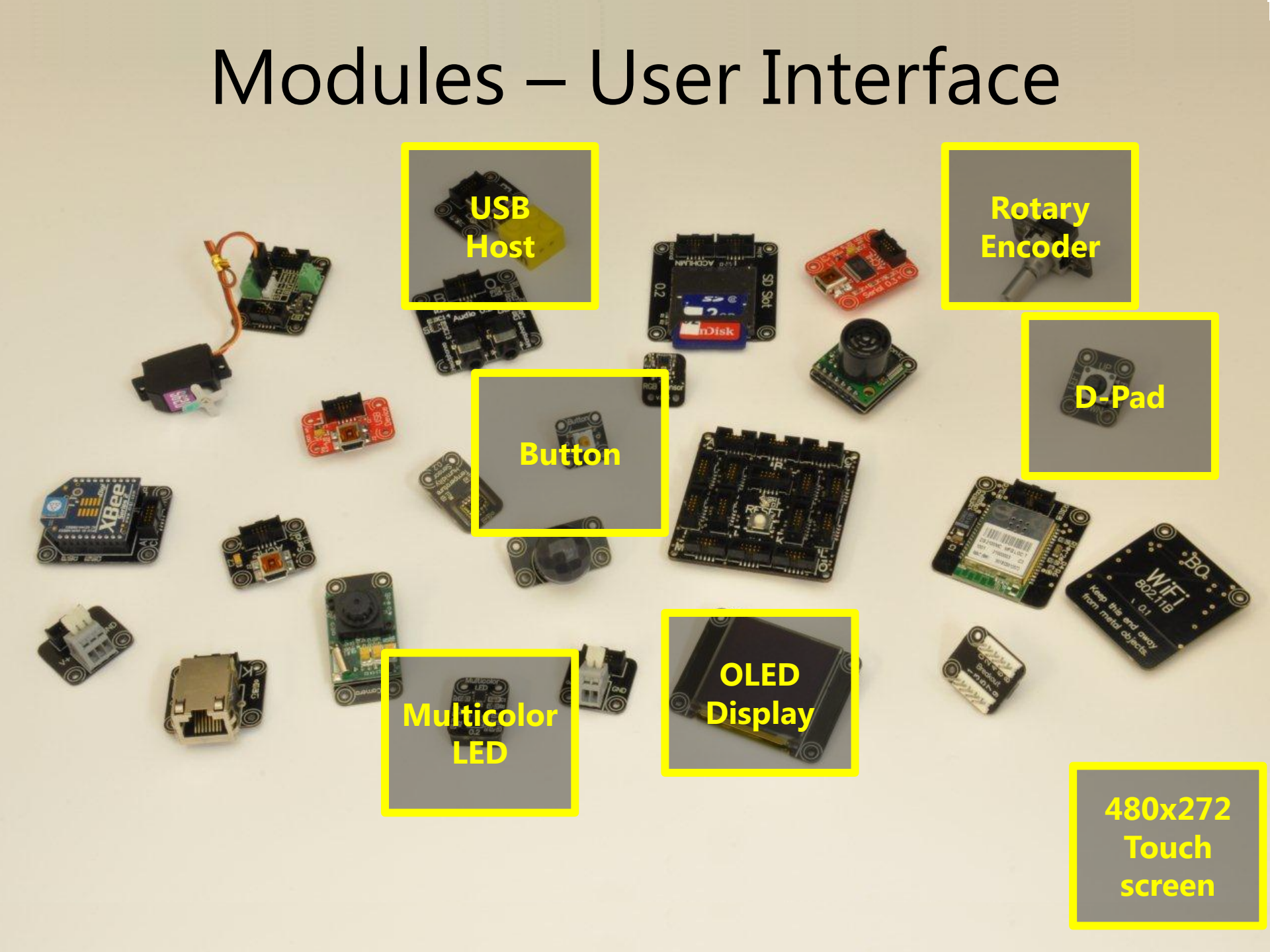
Modules


SD Slot
Audio 0.2
RGB Sensor
XBee
WiFi
802.11B
Keep this end away from metal objects.

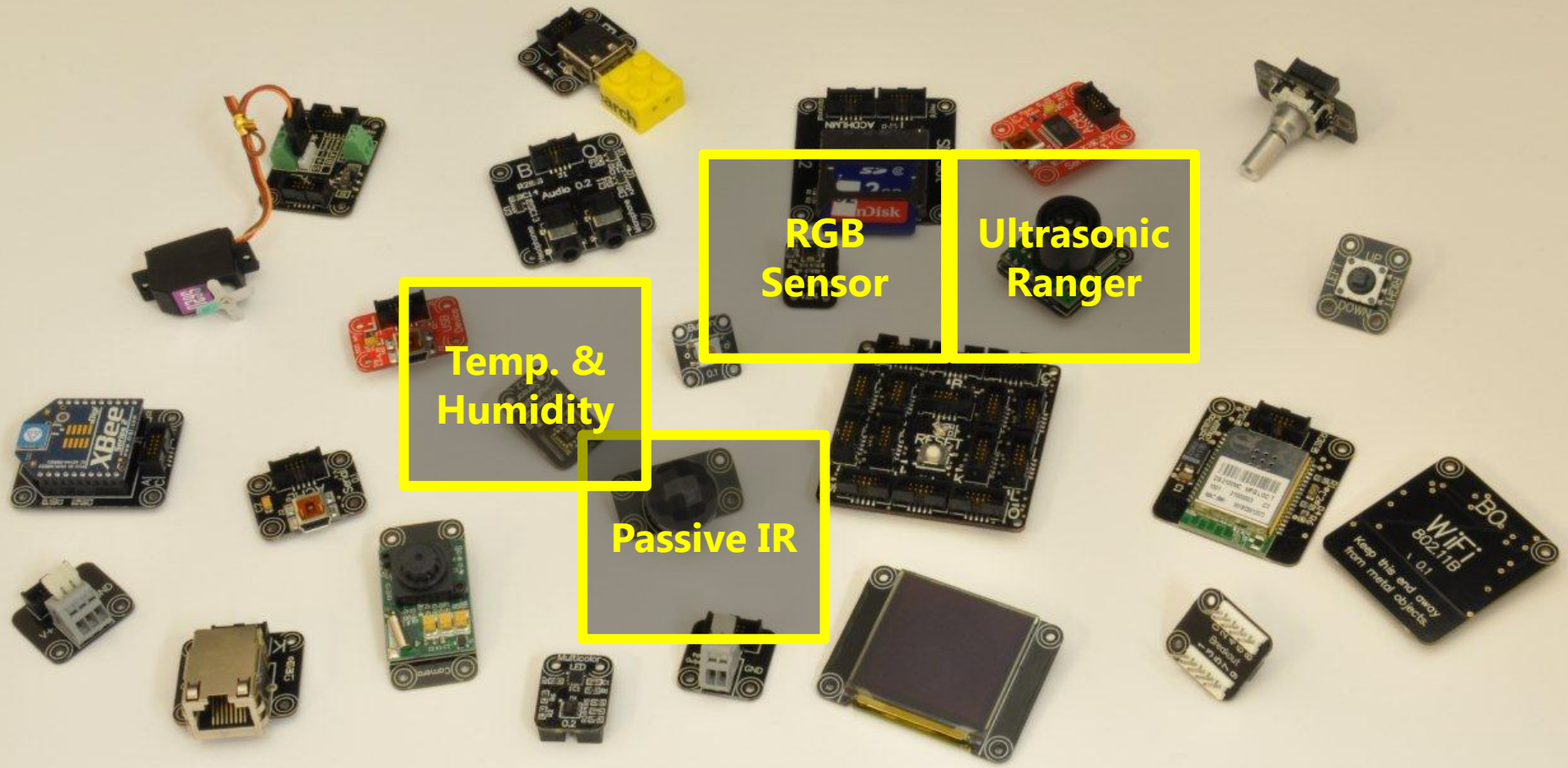
The Mainboard



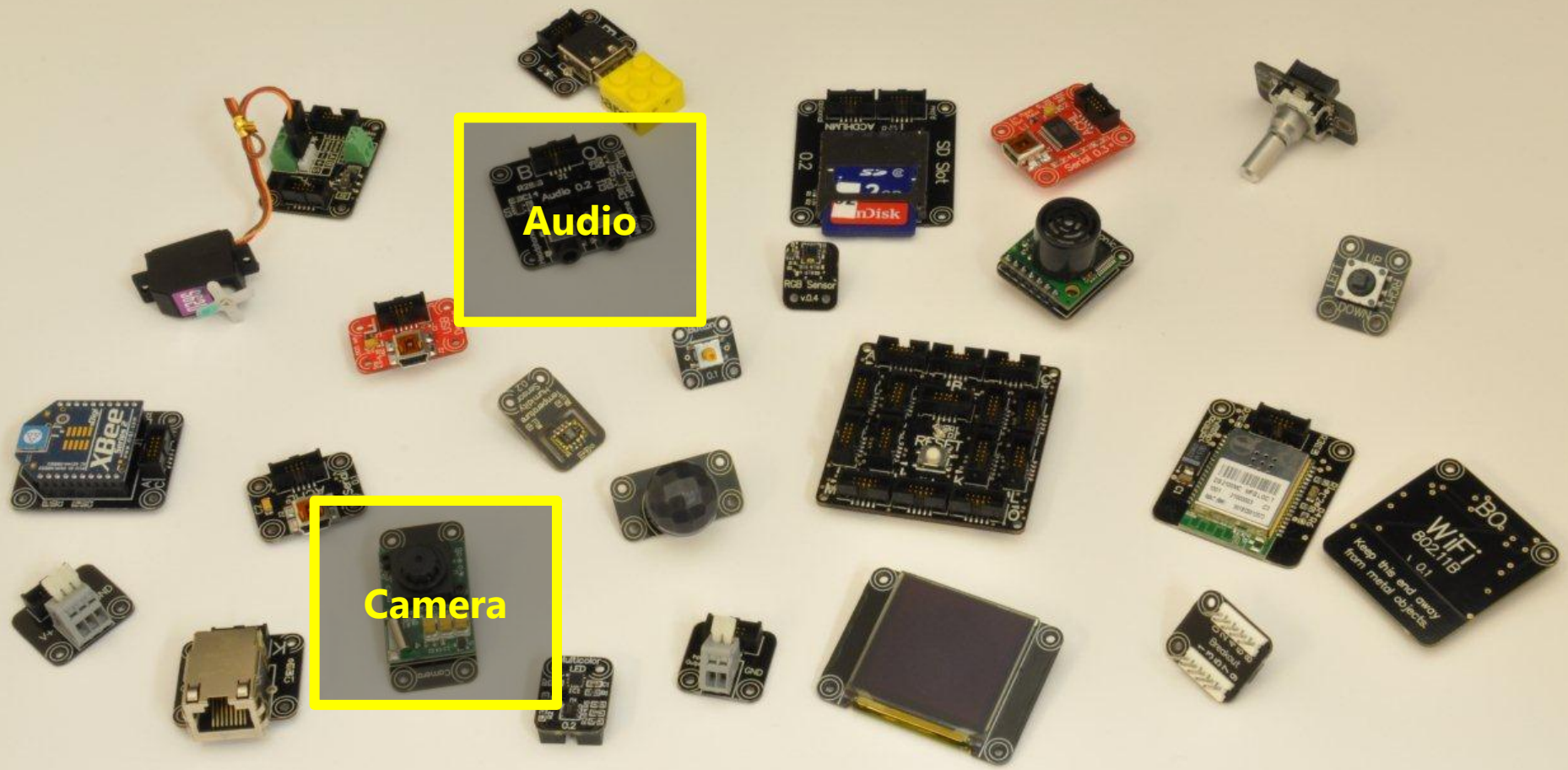
Modules – User Interface



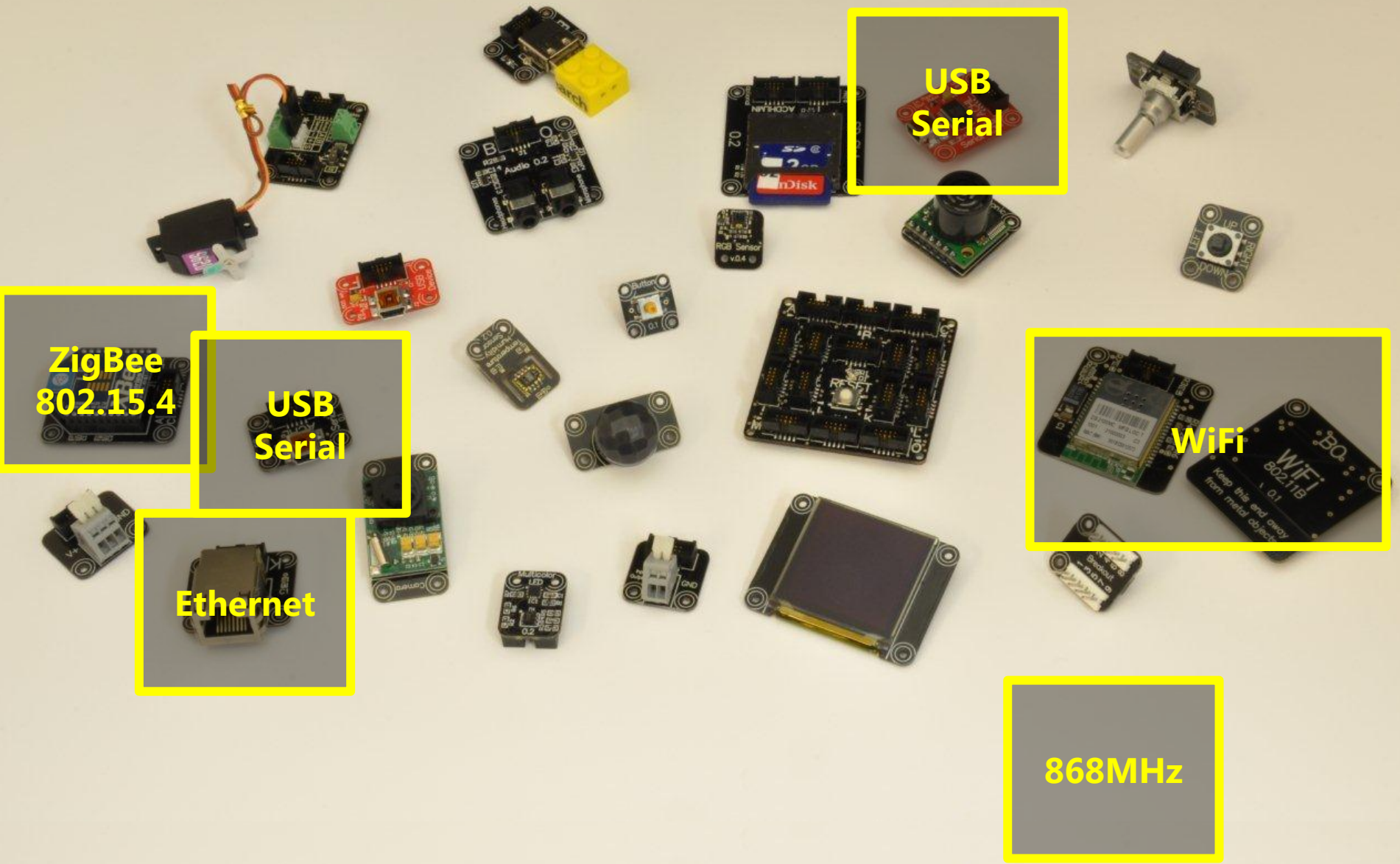
Modules – Sensors



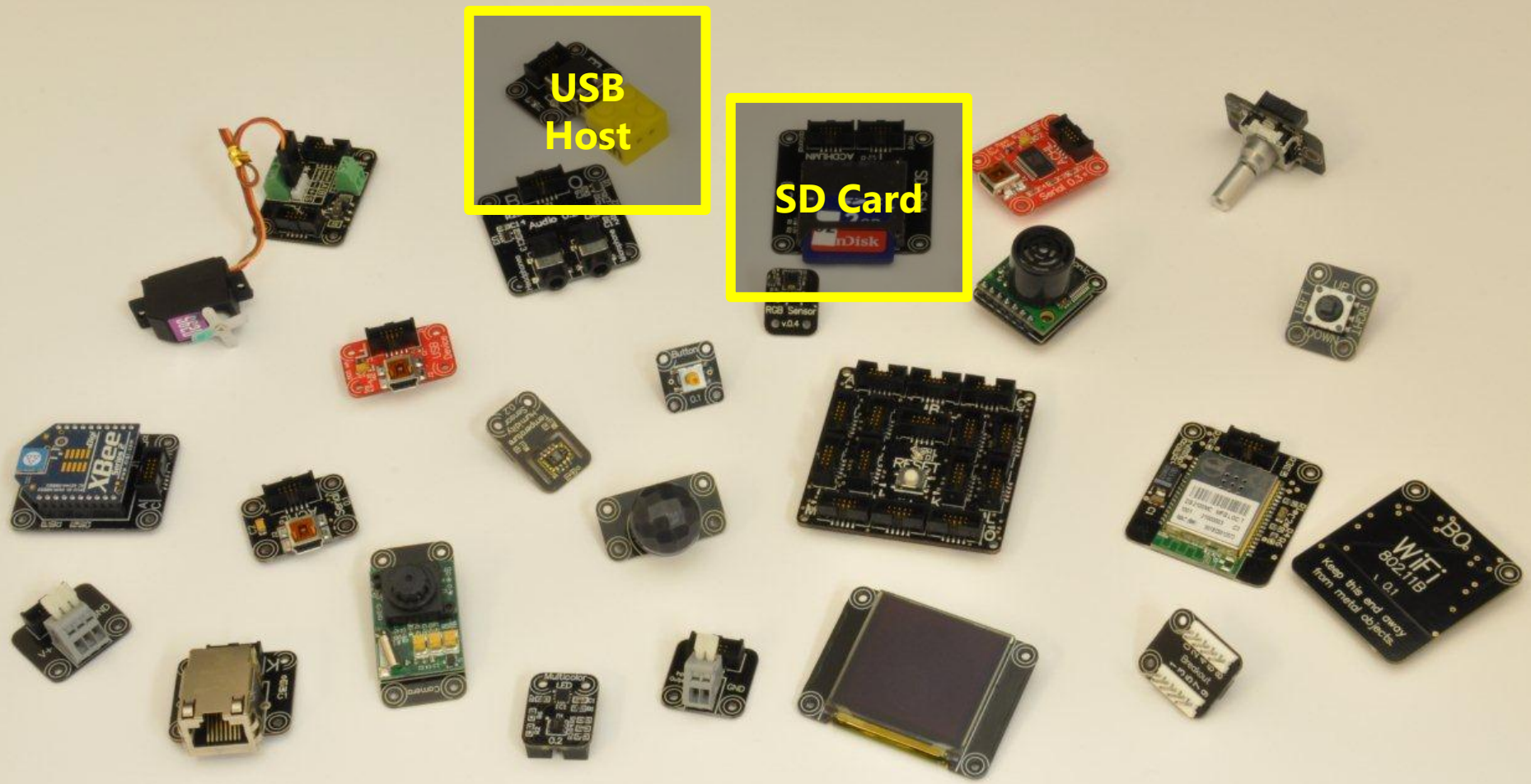
Modules – Multimedia



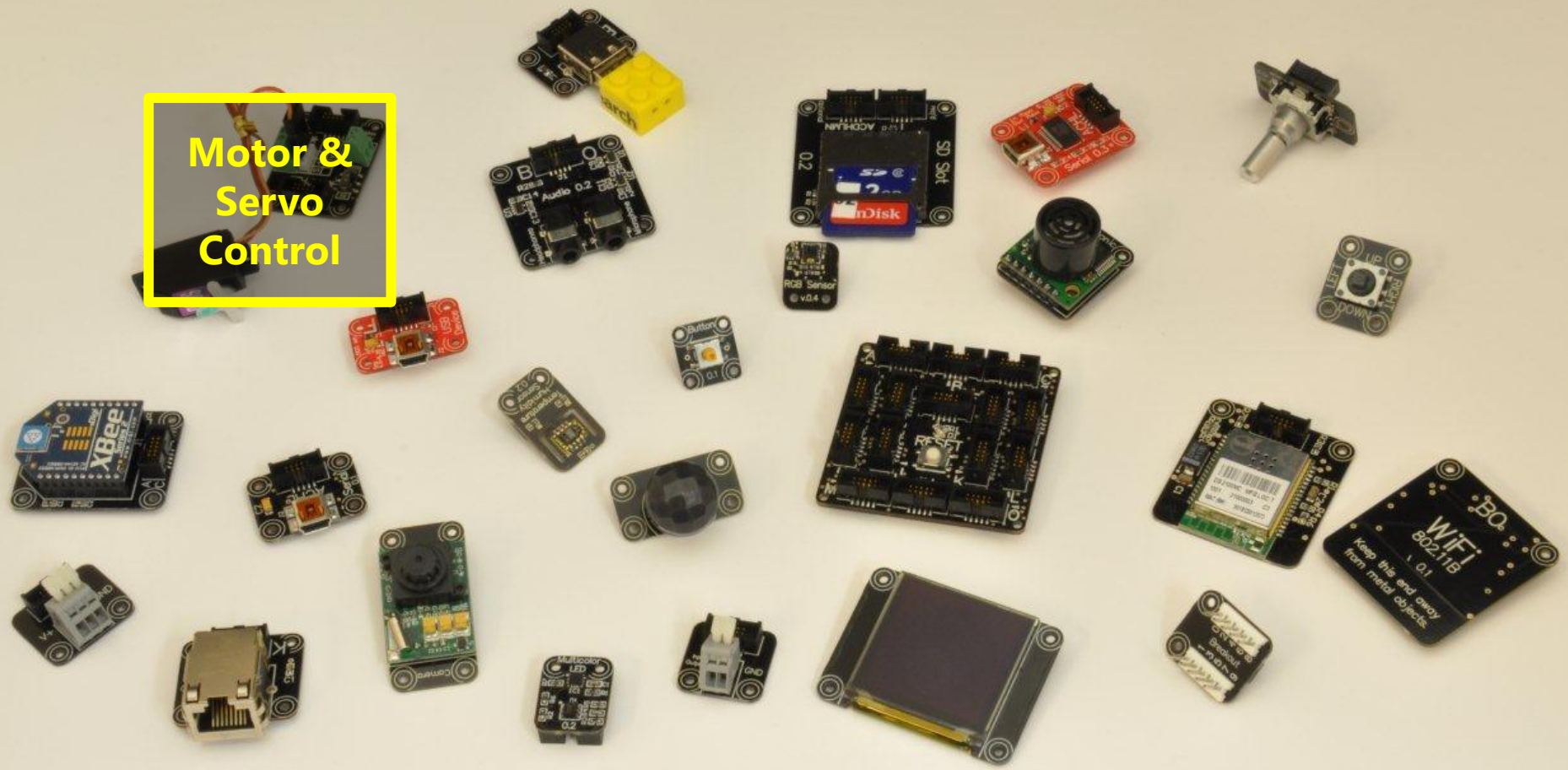
Modules – Networking



Modules – Storage



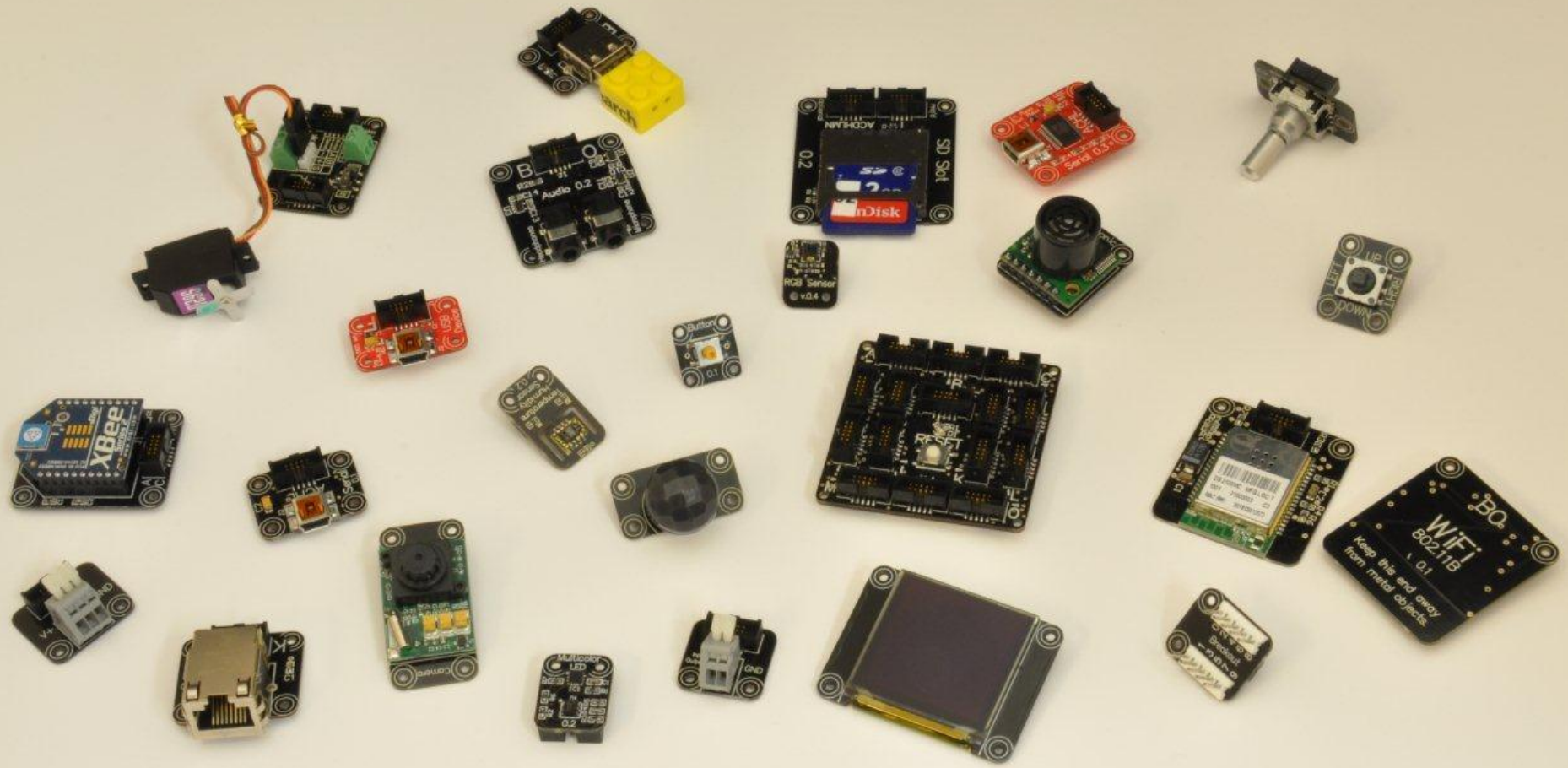
Modules – Actuators



Relay - low voltage

Relay – mains voltage

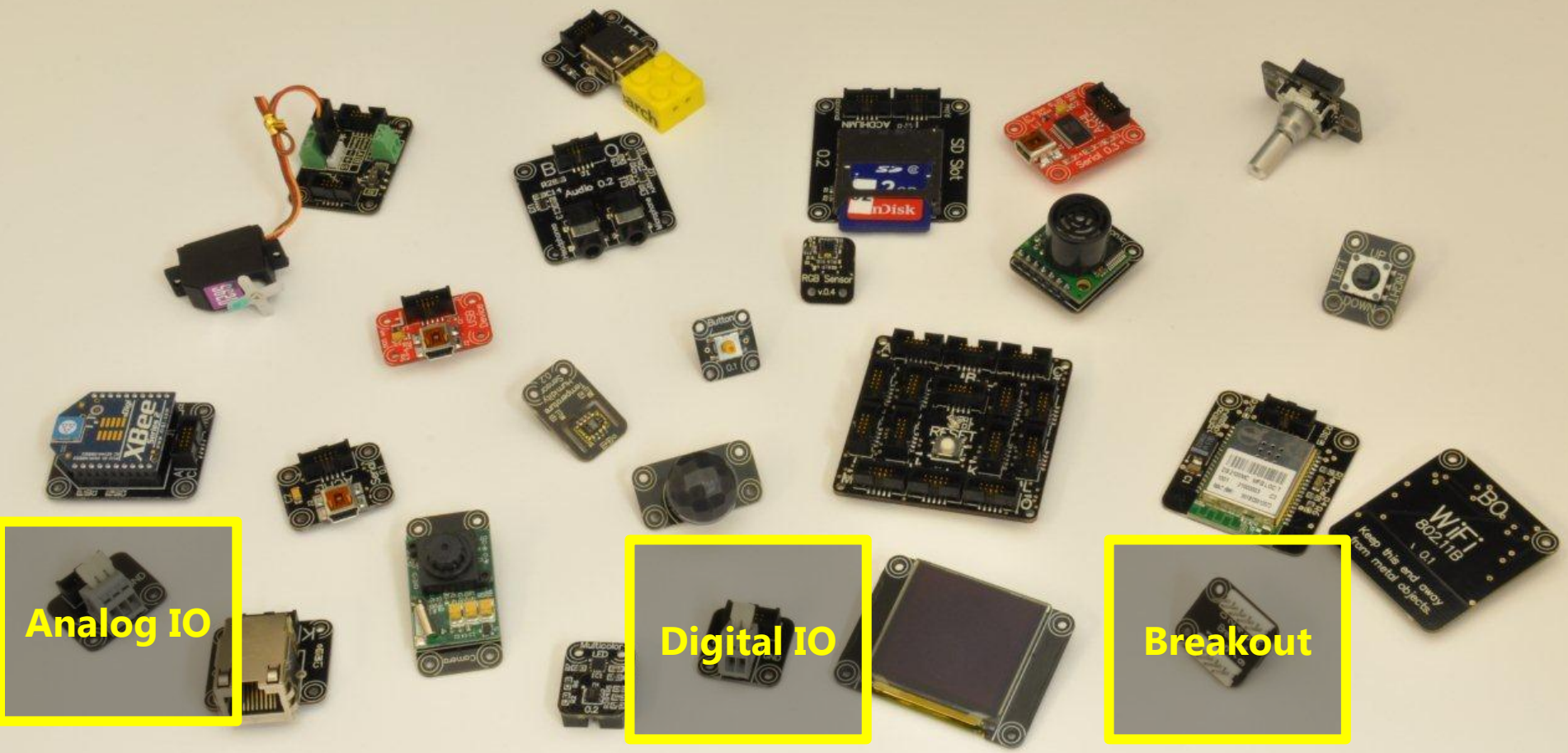
Modules – Power Supply



DC Power

Battery

Modules –Extensibility



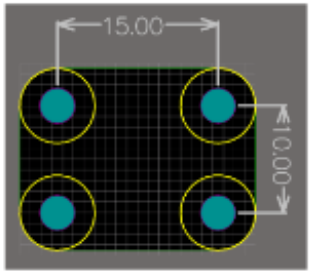
Gadgeteer Socket Pin-Mapping Table

	Basic Socket Types (modules list all that they are compatible with)										
3 GPIO	+3.3V	+5V	GPIO!	GPIO	GPIO	[NC]	[NC]	[NC]	[NC]	GND	X
7 GPIO	+3.3V	+5V	GPIO!	GPIO	GPIO	GPIO	GPIO	GPIO	GPIO	GND	Y
Analog In	+3.3V	+5V	AIN (G!)	AIN (G)	AIN	GPIO	[NC]	[NC]	[NC]	GND	A
CAN	+3.3V	+5V	GPIO!	TD (G)	RD (G)	GPIO	[NC]	[NC]	[NC]	GND	C
USB Device	+3.3V	+5V	GPIO!	D-	D+	GPIO	GPIO	[NC]	[NC]	GND	D
Ethernet	+3.3V	+5V	[NC]	LED1 (OPT)	LED2 (OPT)	TX D-	TX D+	RX D-	RX D+	GND	E
SD Card	+3.3V	+5V	GPIO!	DAT0	DAT1	CMD	DAT2	DAT3	CLK	GND	F
USB Host	+3.3V	+5V	GPIO!	D-	D+	[NC]	[NC]	[NC]	[NC]	GND	H
I2C	+3.3V	+5V	GPIO!	[NC]	[NC]	GPIO	[NC]	SDA	SCL	GND	I
UART+Handshaking	+3.3V	+5V	GPIO!	TX (G)	RX (G)	RTS	CTS	[NC]	[NC]	GND	K
Analog Out	+3.3V	+5V	GPIO!	GPIO	AOUT	[NC]	[NC]	[NC]	[NC]	GND	O
PWM	+3.3V	+5V	GPIO!	[NC]	[NC]	GPIO	PWM (G)	PWM (G)	PWM	GND	P
SPI	+3.3V	+5V	GPIO!	GPIO	GPIO	CS	MOSI	MISO	SCK	GND	S
Touch	+3.3V	+5V	[NC]	YU	XL	YD	XR	[NC]	[NC]	GND	T
UART	+3.3V	+5V	GPIO!	TX (G)	RX (G)	GPIO	[NC]	[NC]	[NC]	GND	U
LCD 1	+3.3V	+5V	LCD R0	LCD R1	LCD R2	LCD R3	LCD R4	LCD VSYNC	LCD HSYNC	GND	R
LCD 2	+3.3V	+5V	LCD G0	LCD G1	LCD G2	LCD G3	LCD G4	LCD G5	BACKLIGHT	GND	G
LCD 3	+3.3V	+5V	LCD B0	LCD B1	LCD B2	LCD B3	LCD B4	LCD EN	LCD CLK	GND	B
Manufacturer Specific	+3.3V	+5V	[MS]	[MS]	[MS]	[MS]	[MS]	[MS]	[MS]	GND	Z
DaisyLink Downstream*	+3.3V	+5V	GPIO!	GPIO	GPIO	[MS]	[MS]	[MS]	[MS]	GND	*

Module Design Guidelines

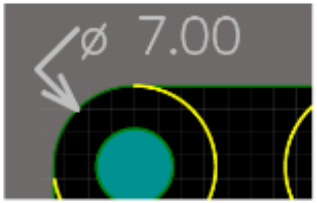
The keep-out area should be clearly delimited in the silkscreen on both sides of the PCB, as shown in the following illustration. For small modules, where space is tight, it is possible to interrupt the keep-out delimiter silkscreen to make space for other labeling or silkscreen elements. Under no circumstances should you place components inside the keep-out area.

All mounting holes should be placed on a 5-mm grid, that is, the distance between adjacent holes should be a multiple of 5 mm, as shown in the following illustration.



Corners

Corners should be rounded, with a 7-mm-diameter curve that is concentric with a mounting hole's keep out area, as shown in the following illustration.



If a corner does not include a mounting hole, the corner does not need to be rounded. However, we recommend that you maintain the same 7-mm rounding diameter for consistency.

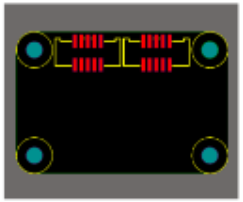
Multiple Connectors

A module might include more than one connector in the following two instances:

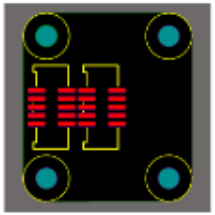
- The module needs to provide connection to more than one mainboard socket at a time.
- The module is compatible with the .NET Gadgeteer DaisyLink Protocol and can be daisy-chained with other modules.

Modules with Multiple Socket Connectors

In this case, connectors can be placed anywhere on the board, following the guidelines under "Connectors" earlier in this document. If possible, connectors should be placed side by side near an edge of the board, as shown in the following illustration.



If the dimensions of the board prohibit a side-by-side arrangement of connectors, the next best option is to place a second connector immediately behind the first, in the same orientation, as shown in the following illustration.



Object-Oriented Programming

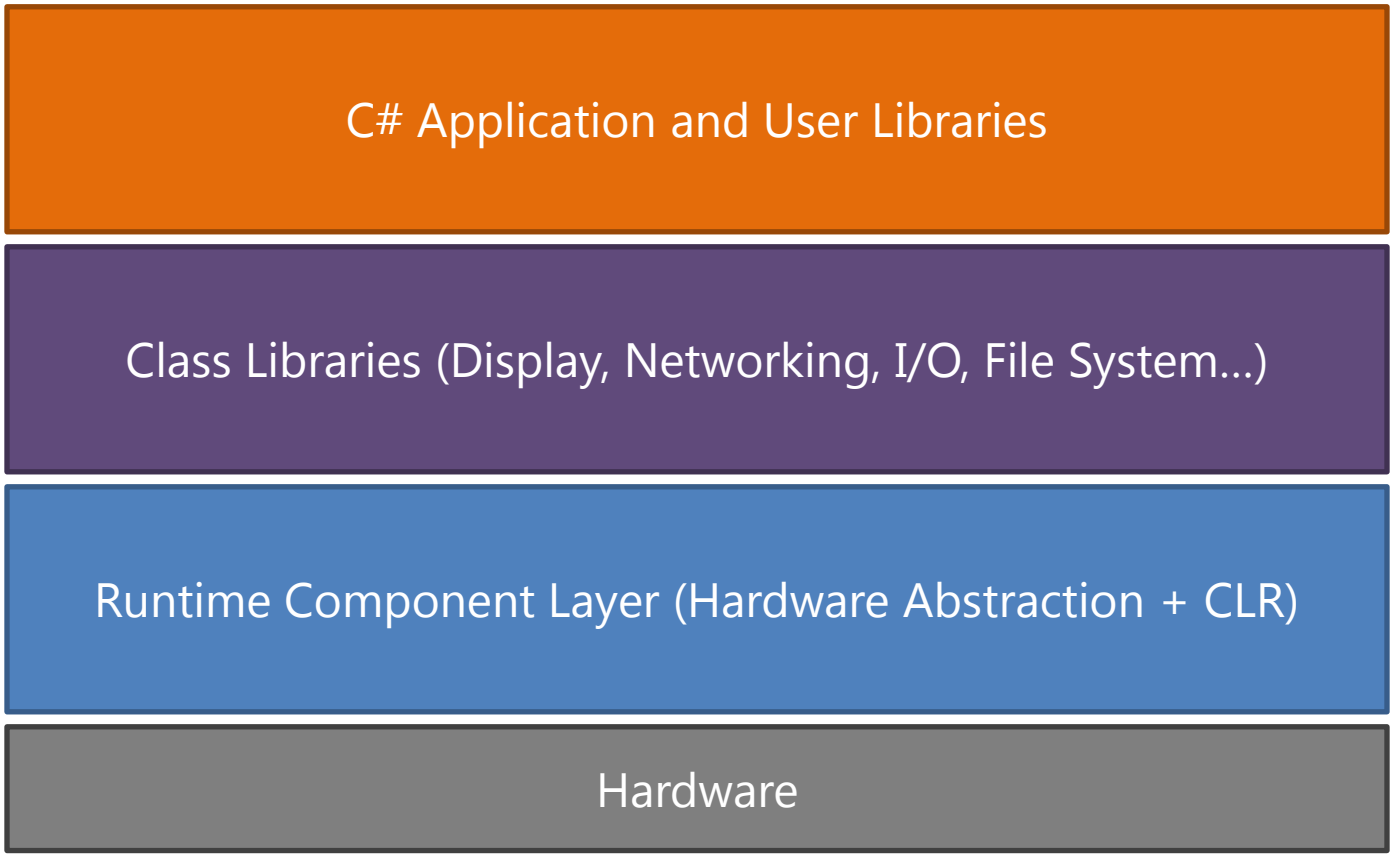
```
void ProgramStarted()
{
    // Initialize GTM.Modules an
    myButton = new GTM.Button(GT
    myLed = new GTM.MulticolorLE

    myButton.

    // Do one
    Debug.Pri
}
```

- ButtonPressed
- ButtonReleased
- DebugPrintEnabled
- Equals
- GetHashCode
- GetType
- IsPressed
- ToString

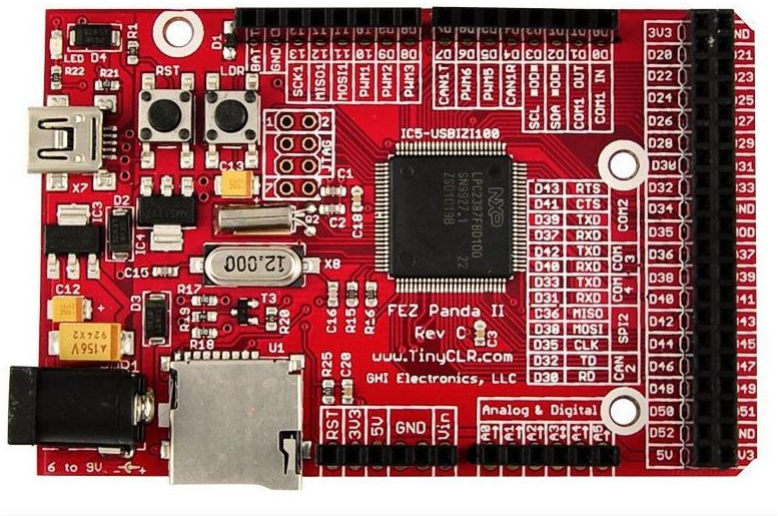
Software Development Libraries

- Gadgeteer uses the Microsoft .NET Micro Framework (NETMF), which provides a simple and powerful way to write software for small devices
- Software is developed and debugged in Visual Studio, and code is in managed, object-oriented C#
- The SDK provides classes encapsulating functionality for individual hardware modules as well as other utility functions

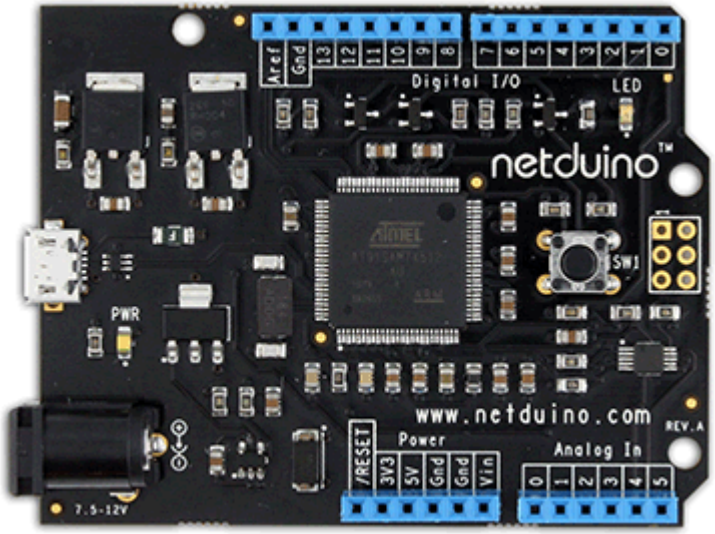
.NET Micro Framework



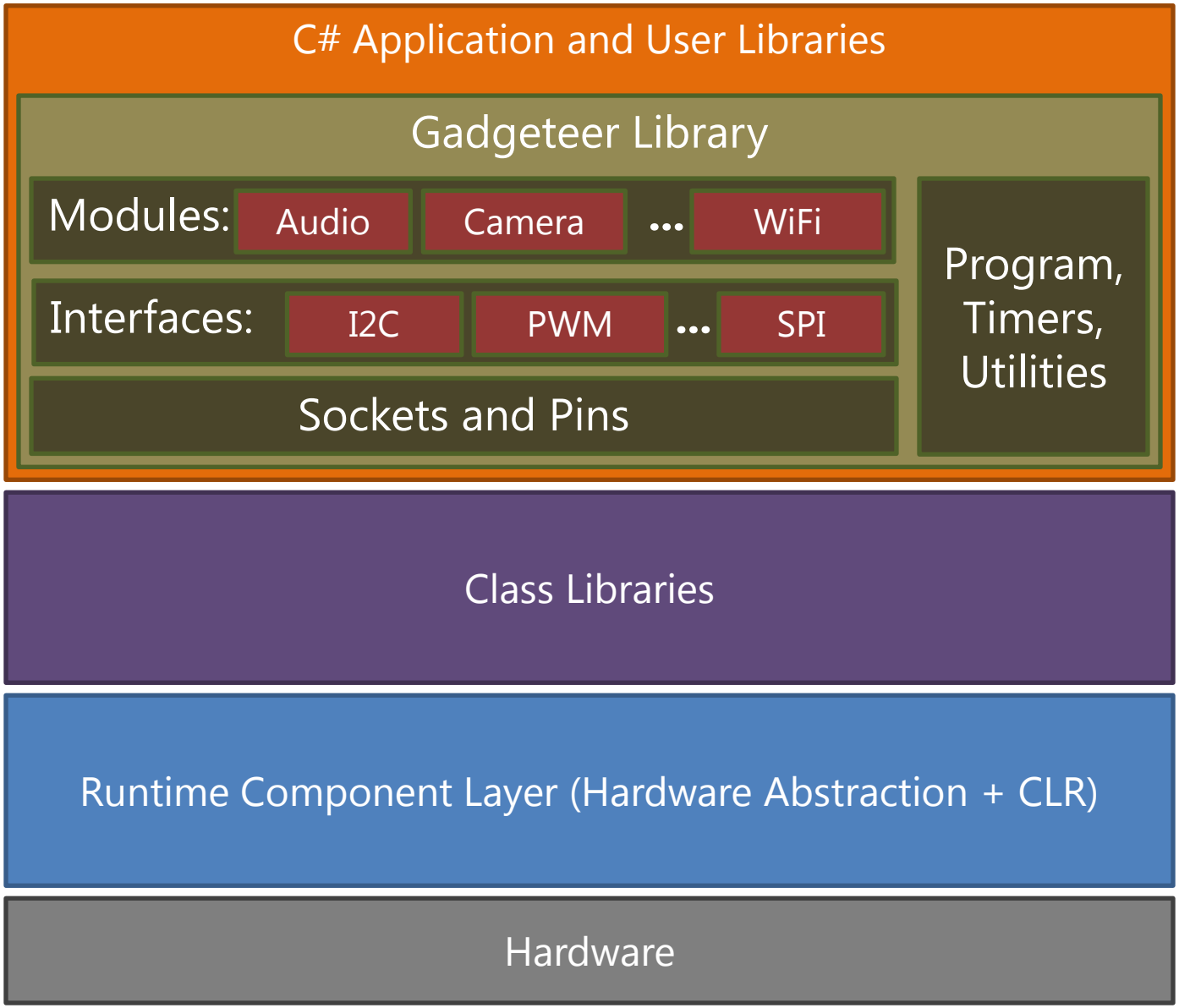
Other NETMF devices



FEZ (GHI Electronics)



Netduino (Secret Labs)


C# Application and User Libraries
Gadgeteer Library
Modules:
Audio
Camera
...
WiFi
Interfaces:
I2C
PWM
...
SPI
Sockets and Pins
Program, Timers, Utilities
Class Libraries
Runtime Component Layer (Hardware Abstraction + CLR)
Hardware

Coding aids

```
public partial class Program
{
    // Define GTM.Modules here
    GTM.
    void
    {
    }
}
```

AnalogInput
Audio
Breakout
Button
Camera
DaisyLinkModule
Deprecated
DigitalIO
Display_128x128

class Microsoft.Gadgeteer.Modules.Button
Represents a button with one of two states, e

Namespace alias to make list of modules easy to find

```
void ProgramStarted()
{
    // Initialize GTM.Modules and event handlers here.
    myButton = new GTM.Button(GTM.Button.CompatibleSocket.D);
    myLed = new GTM.MulticolorLED(GTM.MulticolorLED.CompatibleSocket.L);

    myButton.
    // Do one
    Debug.Pri
}
```

ButtonPressed
ButtonReleased
DebugPrintEnabled
Equals
GetHashCode
GetType
IsPressed
ToString

GTM.Button.ButtonEventHandler Button.ButtonPressed
Raised when the Microsoft.Gadgeteer.Modules.Button is presse

Enums to provide clear lists of parameter options

Unabbreviated names

In-line help

Thread safe event model

```
myButton.ButtonPressed +=
```

new GTM.Button.ButtonEventHandler(myButton_ButtonPressed); (Press TAB to insert)

Auto-generated event handlers

Coding aids

Include many parameters since local vars are easier

```
void button_ButtonPressed(GIM.Button sender, GIM.Button.ButtonState state)
{
    Console.WriteLine("Button Pressed");
}
```

High level object-oriented API

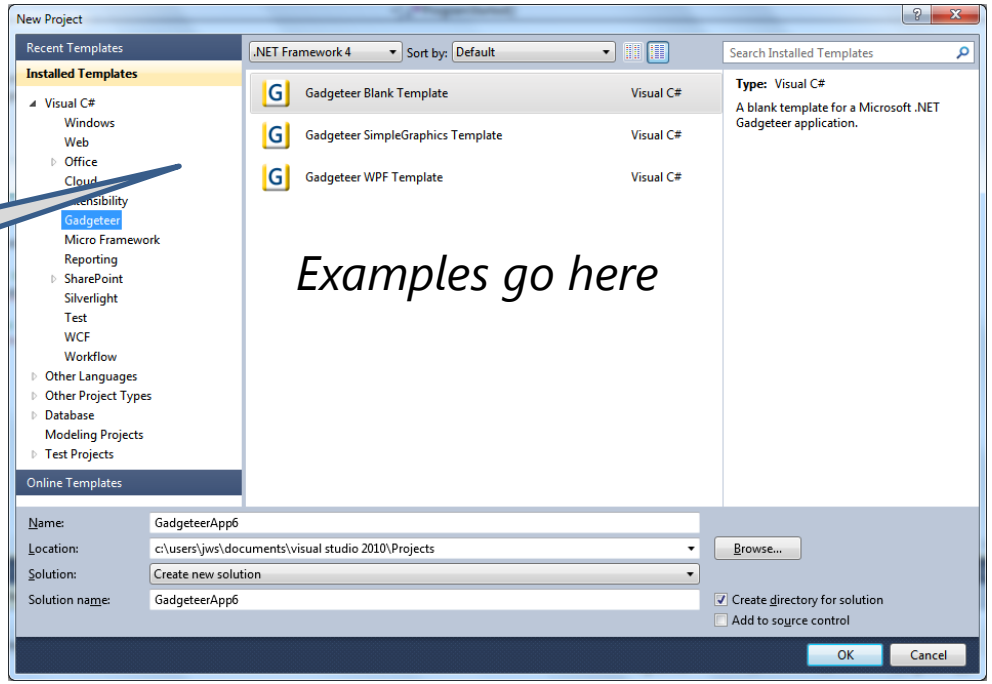
Provide simple *and* complex APIs when appropriate

```
display.SimpleGraphics.DisplayImage(
▲ 3 of 6 ▼ void SimpleGraphics.DisplayImage(string filePath, Bitmap.BitmapImageType imageType, uint x, uint y)
    Displays an image from a file on the screen.
    filePath: The path to the image file.
```

Wrappers enabling fewer library calls

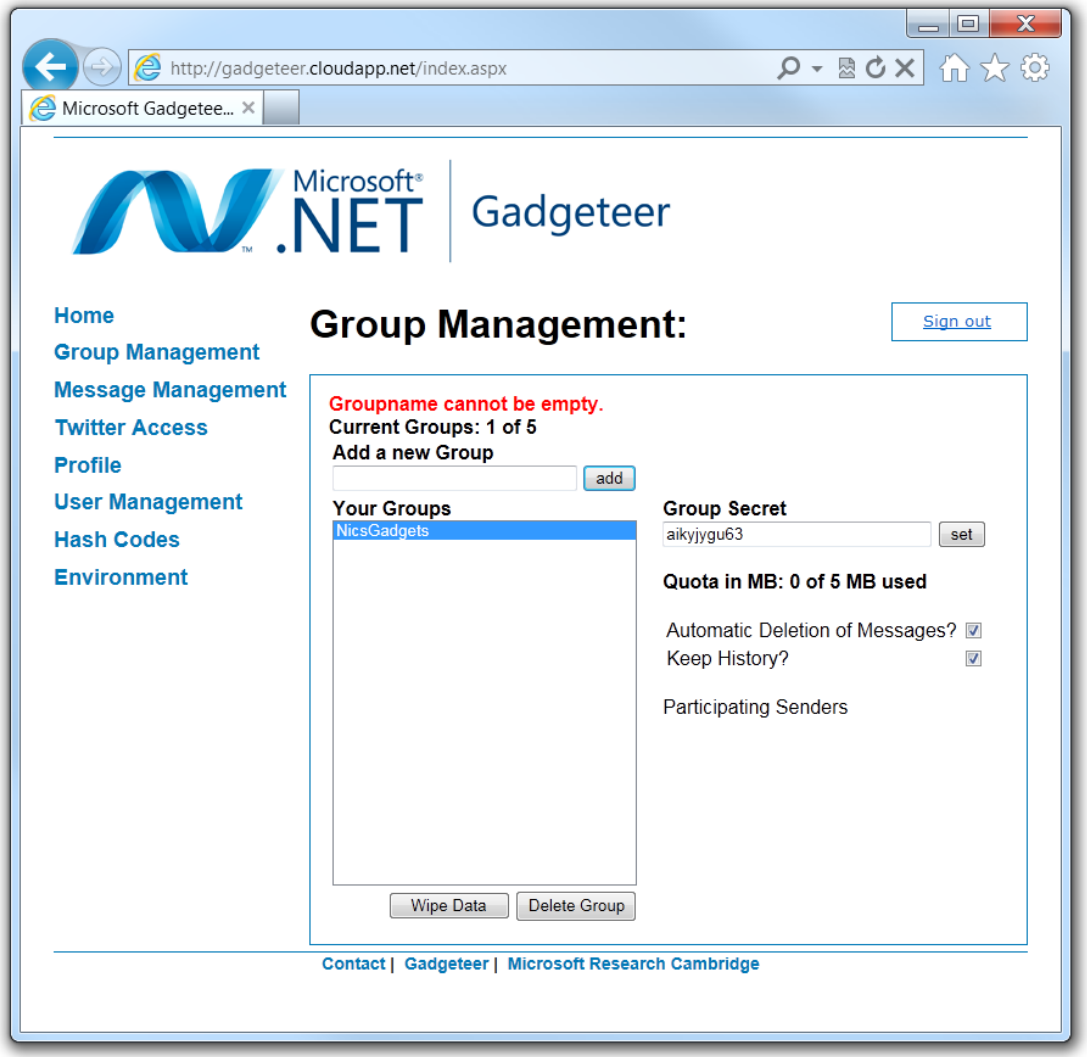
Many method signatures avoiding need to specify default values

Templates and examples



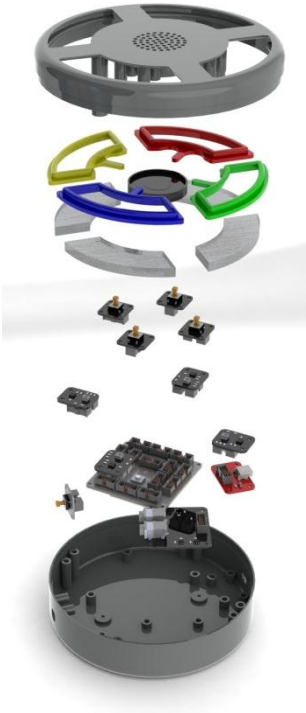
Gadgeteer Cloud Portal

For Web-Of-Things Apps

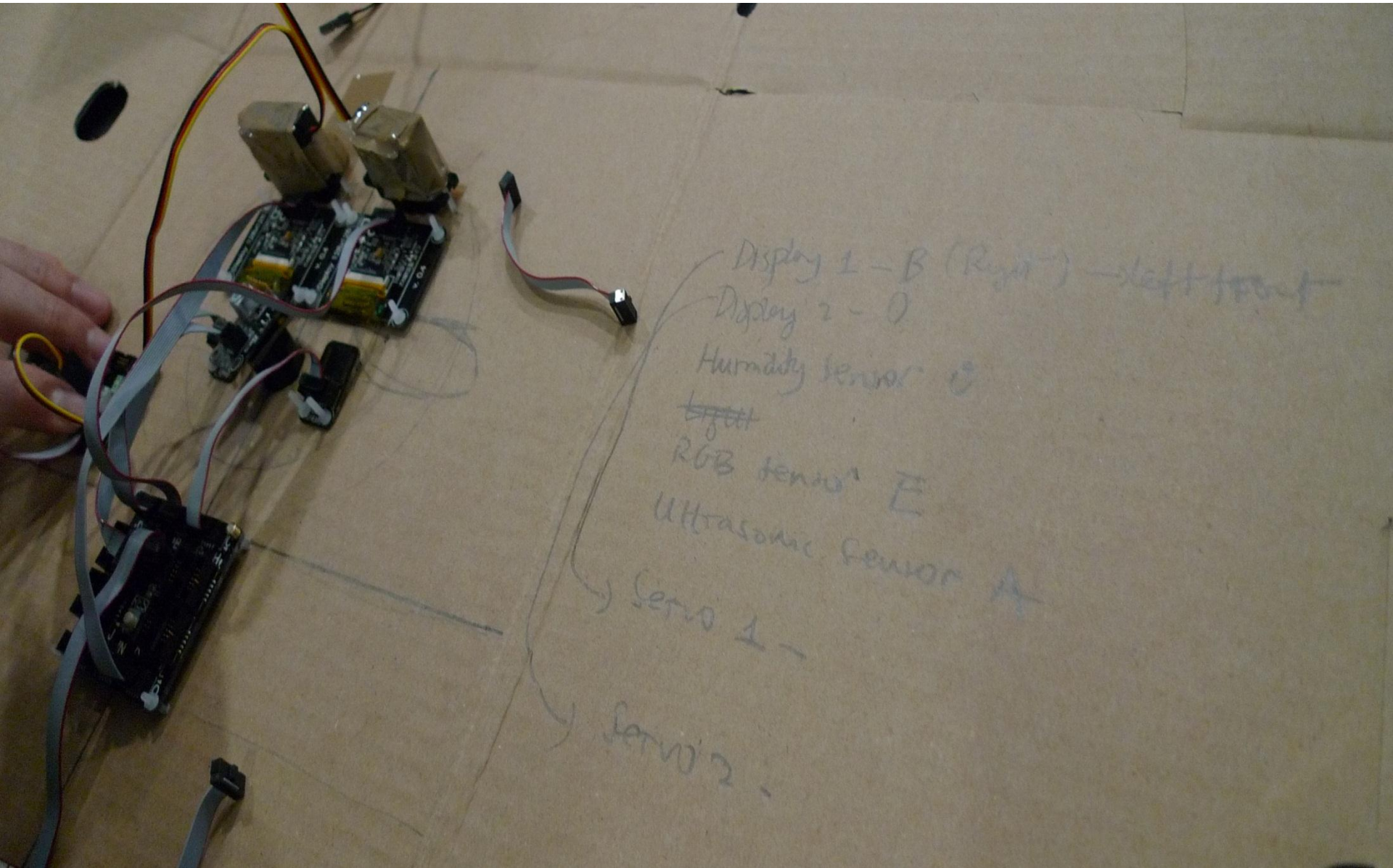



Other Web Services (e.g. Twitter)
Gadgeteer Cloud Portal (Azure Service)
Windows 7

Digital Design and Fabrication



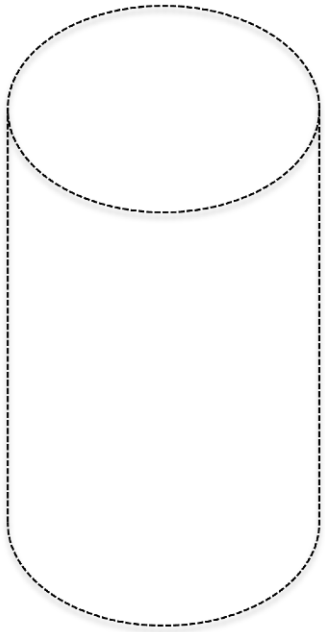
Cardboard prototyping



Cardboard prototyping

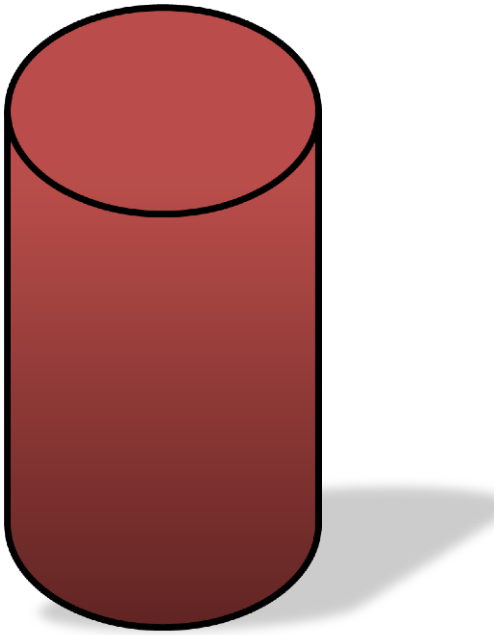
Digital design and rapid manufacture

Digital Design



3D Printer

Physical Object

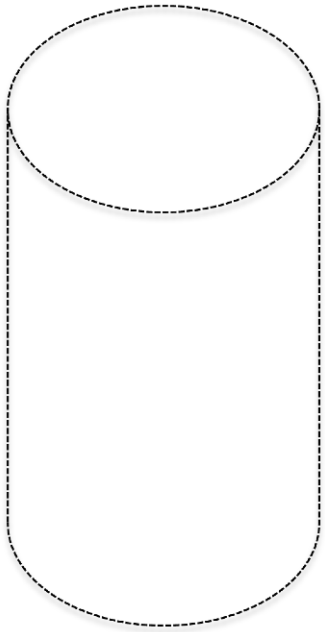


Falling cost and increasing availability of 3D printers

$100,000 $1000

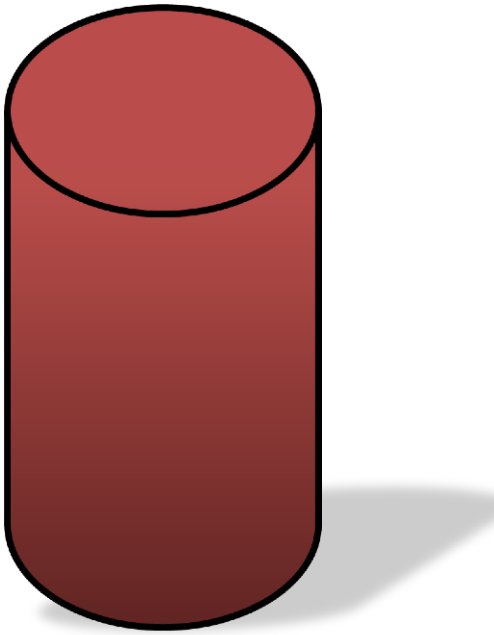
Digital design and rapid manufacture

Digital Design



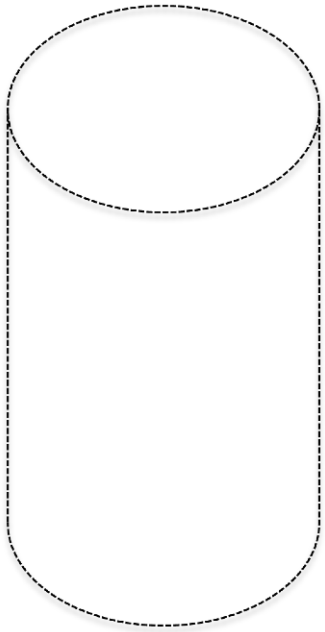
3D Printer

Physical Object



Digital design and rapid manufacture

Digital Design

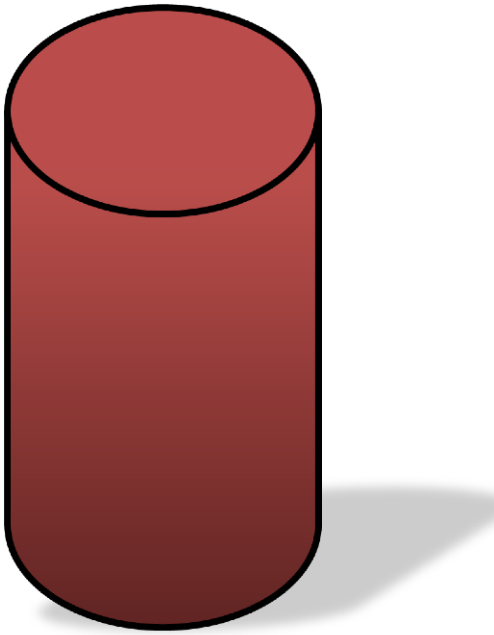


Online 3D Printing Service

shapeways

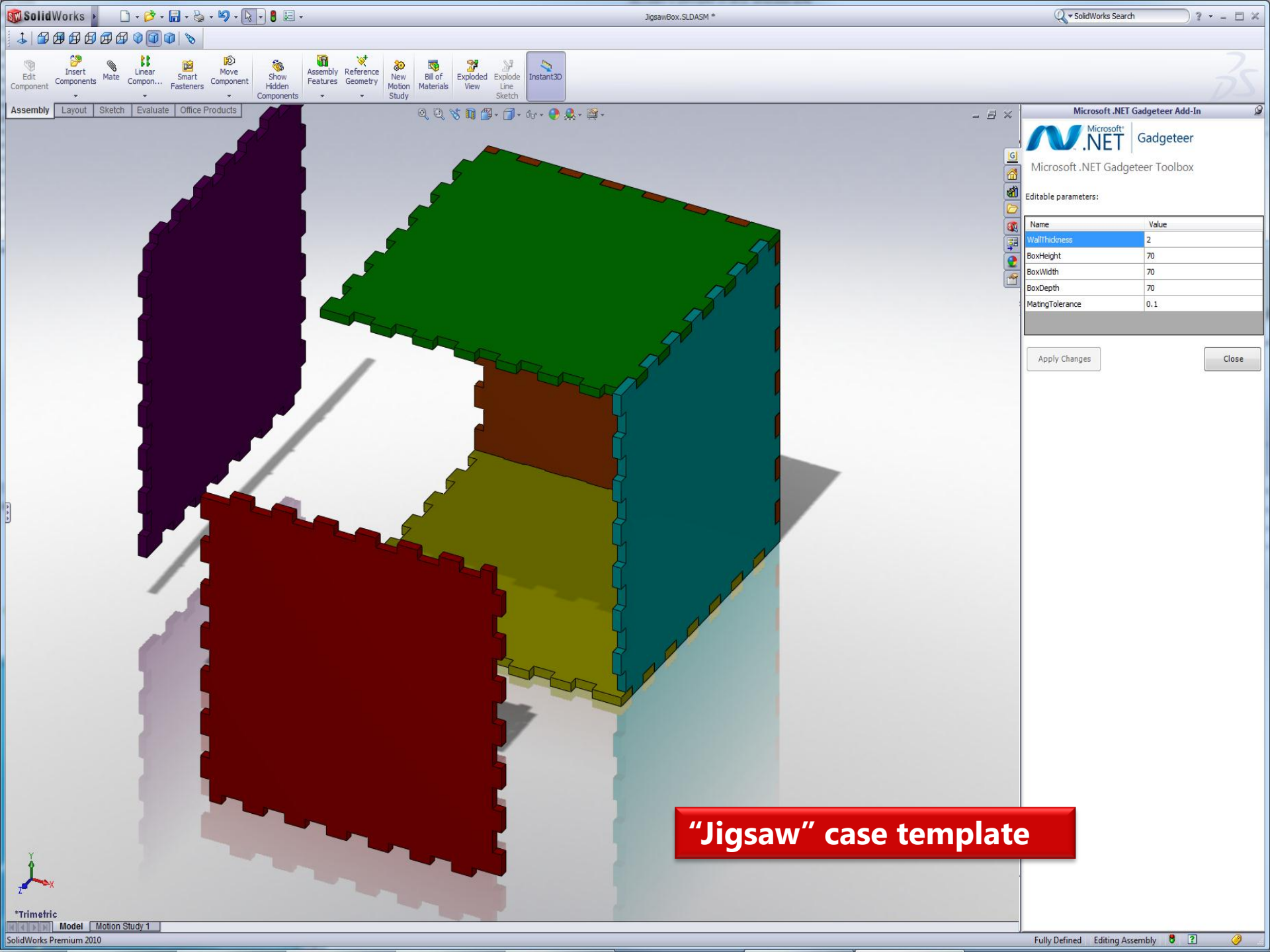
i.materialise

Physical Object

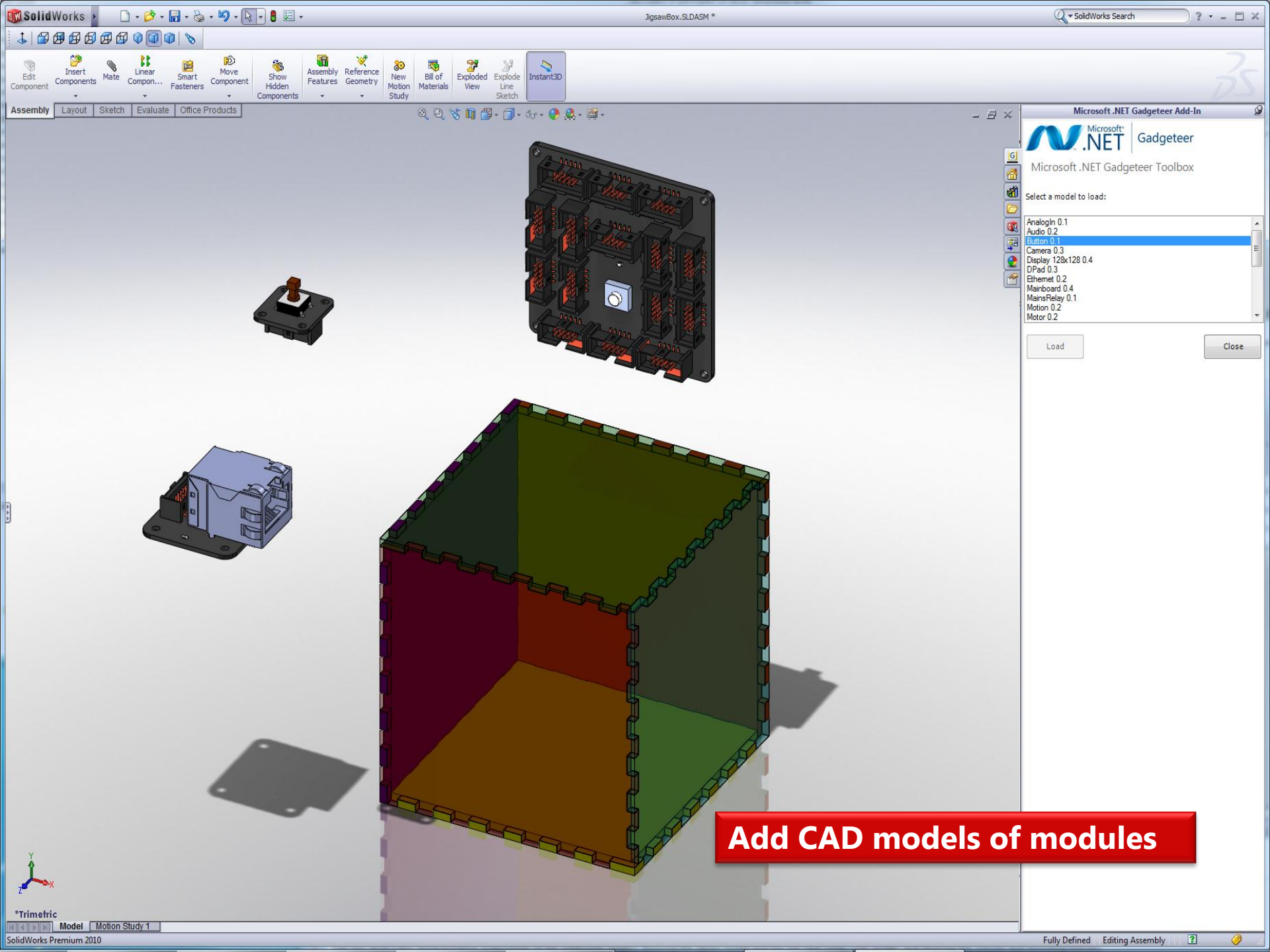


- We want to make it easier to **give shape** to Gadgeteer devices by using digital fabrication technologies

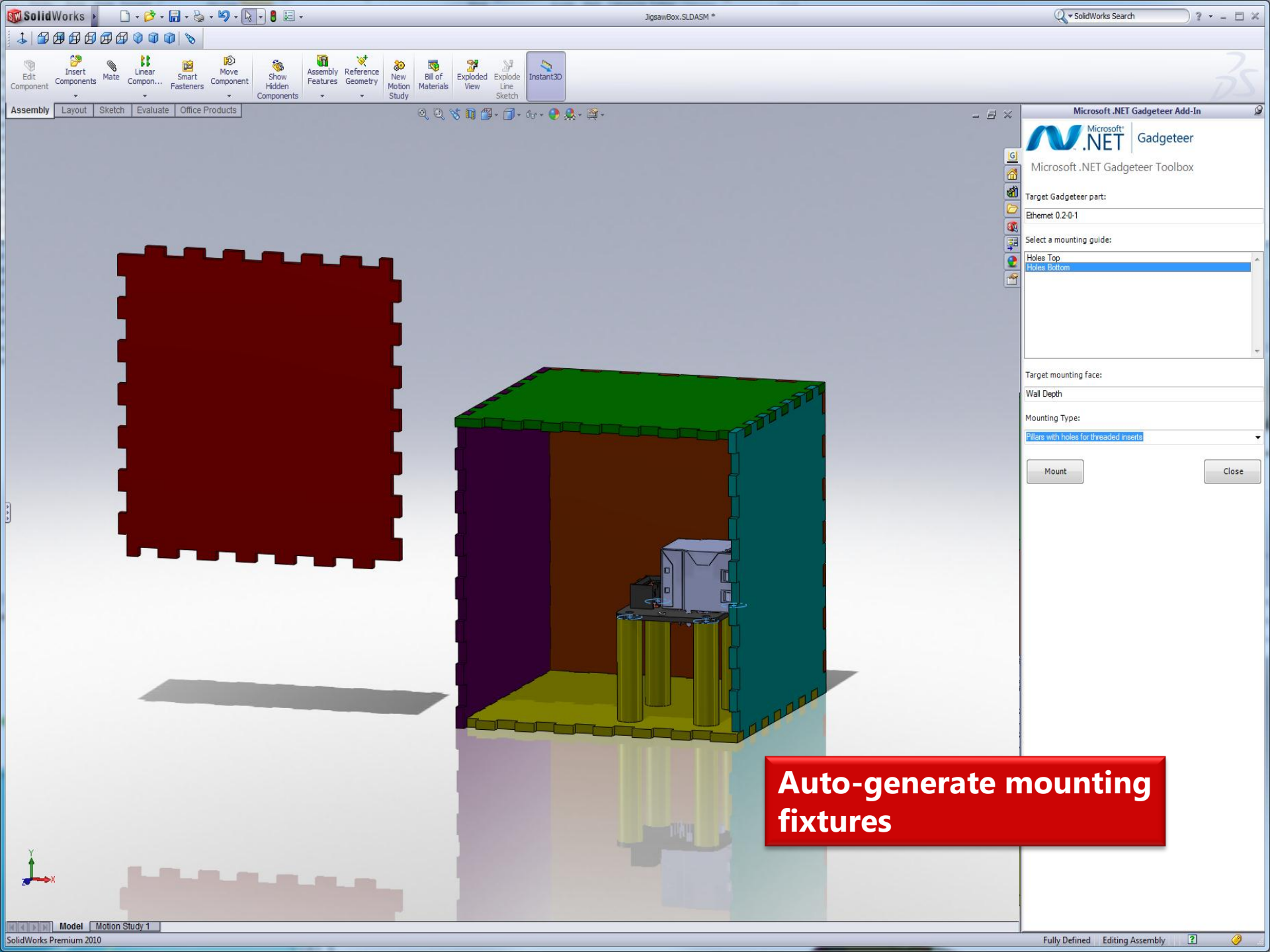
- First step: integration with 3D CAD modelling software (e.g. Solidworks)



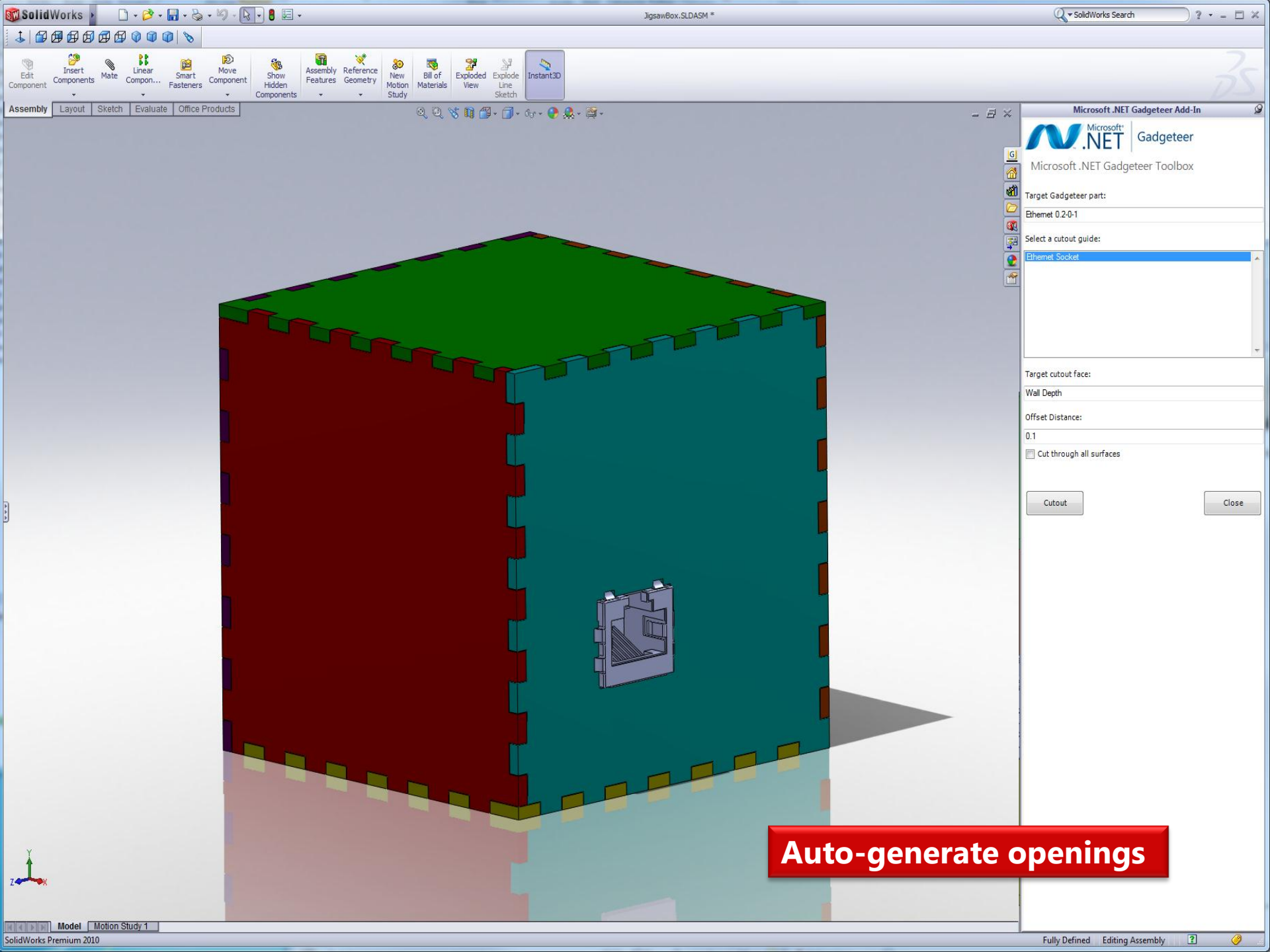
“Jigsaw” case template



Add CAD models of modules



Auto-generate mounting fixtures

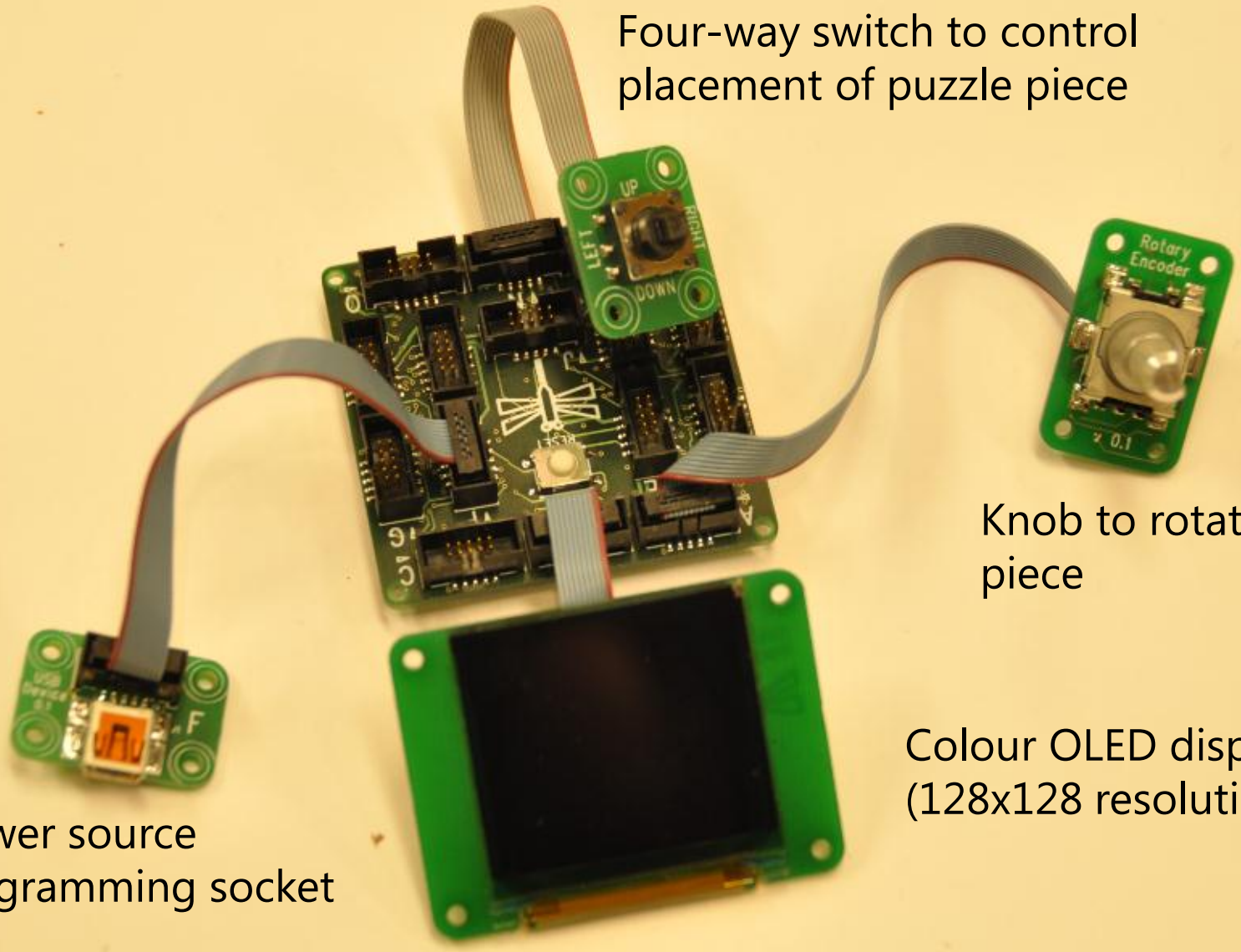


Auto-generate openings

A 24 Hour Prototyping Exercise:

Making a Hand-Held Videogame

Hardware configuration (~5 minutes to assemble)



Four-way switch to control placement of puzzle piece

Knob to rotate piece

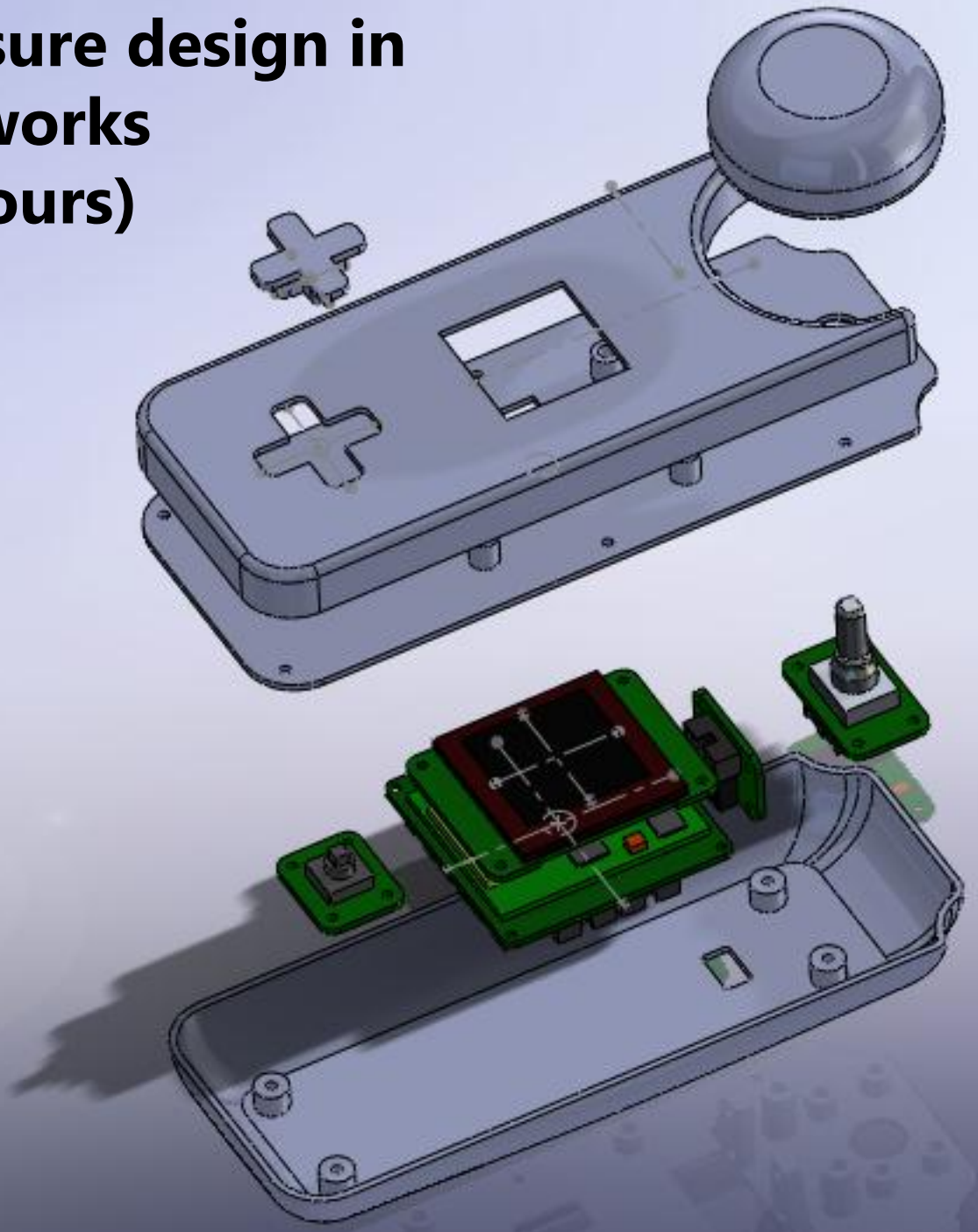
Colour OLED display (128x128 resolution)

USB power source and programming socket

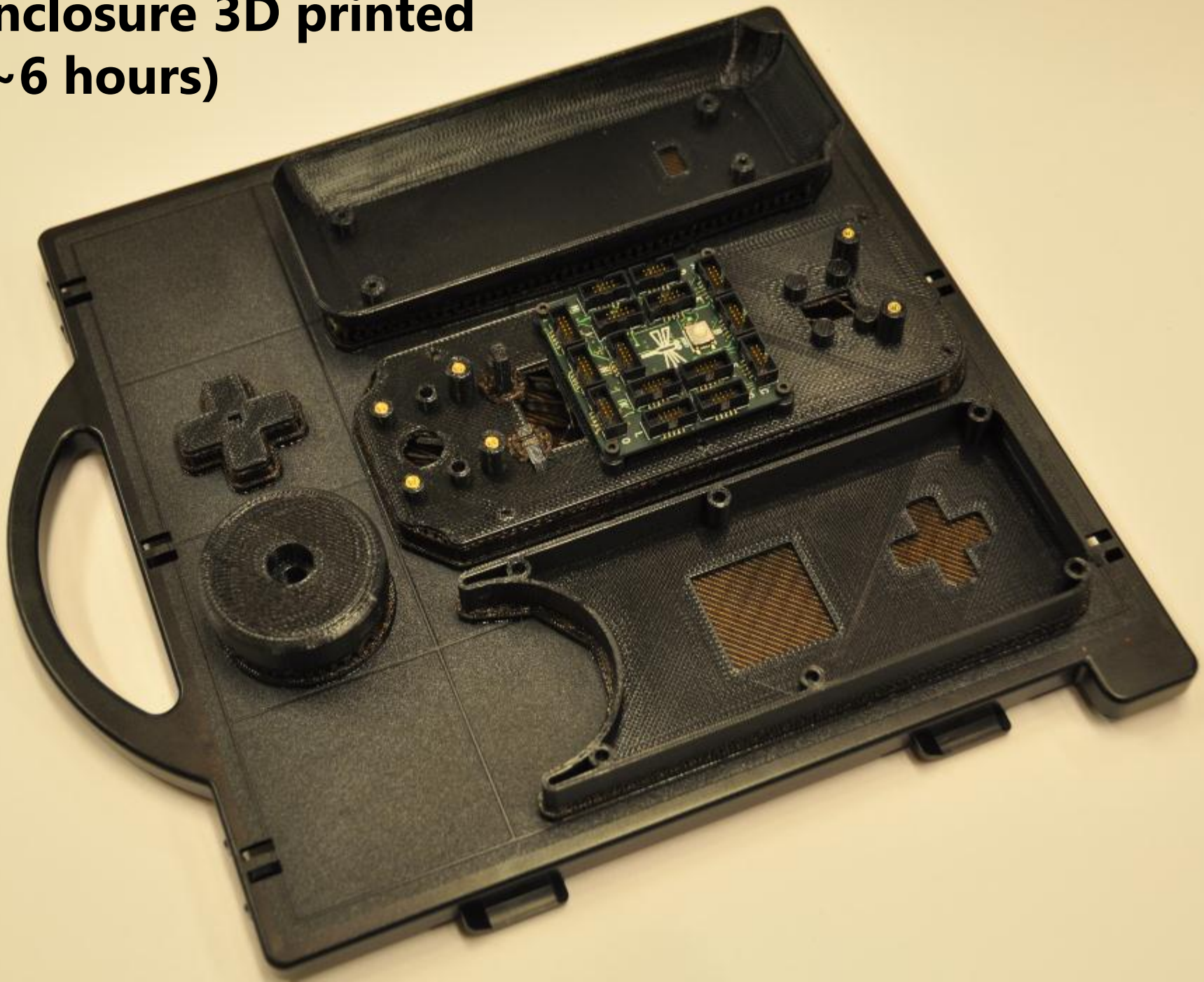
Software development in C# (~5 hours)

```
public class Piece
{
    public Point[] positions;
    public Point displacement;
    public Color color;

    public Piece(Point[] positions, Point displacement, Color color)
    {
        this.positions = positions;
        this.displacement = displacement;
        this.color = color;
    }

    public void Rotate(bool clockwise)
    {
        for (int i = 0; i < positions.Length; i++)
        {
            Point oldpos = positions[i];
            positions[i].x = clockwise ? -oldpos.y : oldpos.y;
            positions[i].y = clockwise ? oldpos.x : -oldpos.x;
        }
    }

    public Piece Clone()
    {
        Piece clone = new Piece((Point[])positions.Clone(), new Point(displacement.
        return clone;
    }

}
```

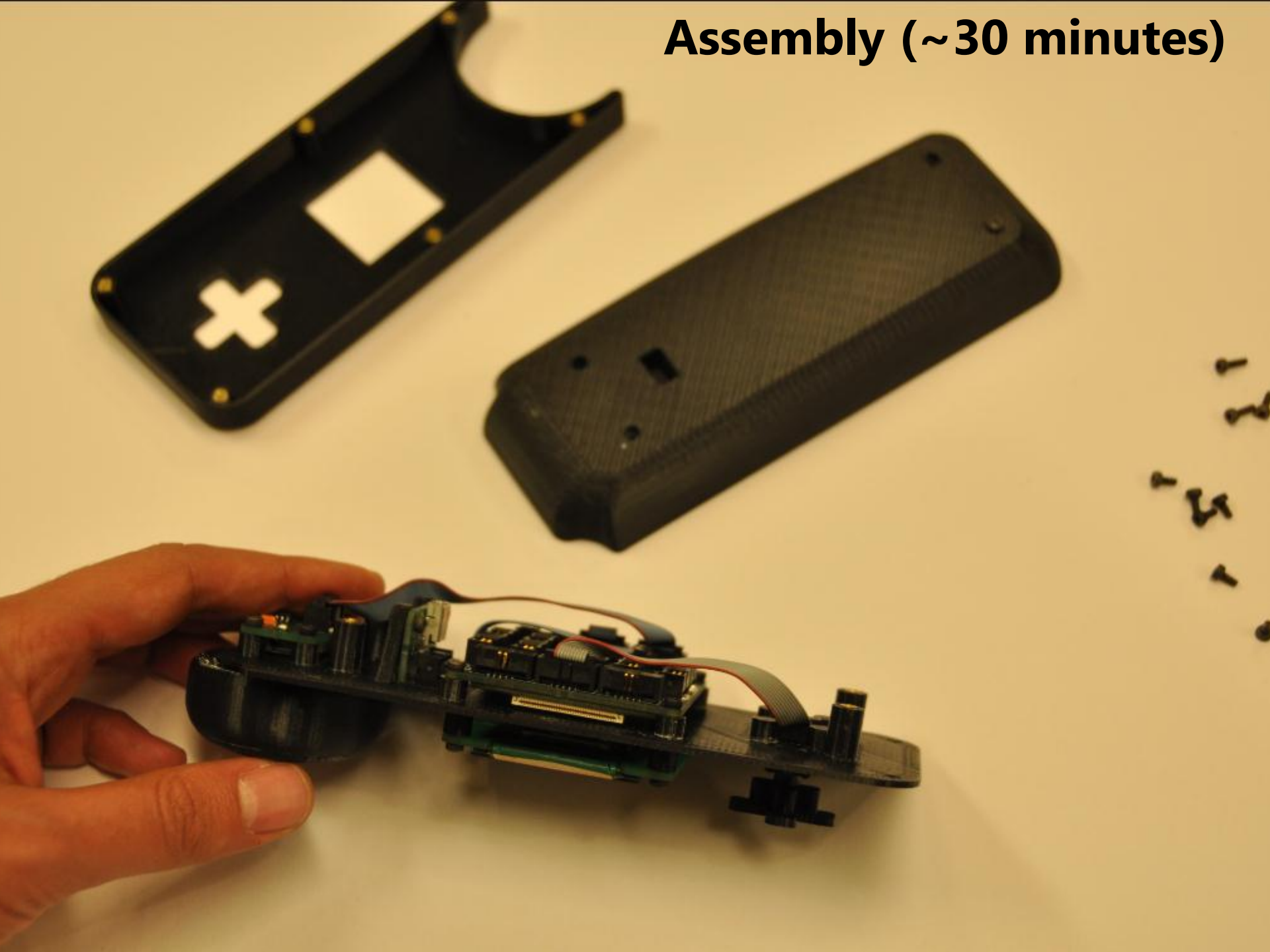
Enclosure design in Solidworks (~3 hours)



Enclosure 3D printed (~6 hours)



Assembly (~30 minutes)



Next steps: Getting .NET Gadgeteer out of the lab

- .NET Gadgeteer software, hardware design and design guidelines released as open source project:
  http://gadgeteer.codeplex.com/

- Community site (in development):
  http://netmf.com/gadgeteer

Next steps: Getting .NET Gadgeteer out of the lab

- Working with a number of hardware manufacturers who will build, distribute and sell the hardware modules
- Initial availability expected end of July
- Started kit priced around $250
- More modules to become available from different manufacturers during the rest of the year

More information

Please get in touch if you are interested in using .NET Gadgeteer for research or teaching

gadgeteer@microsoft.com

Microsoft® .NET™ | Gadgeteer