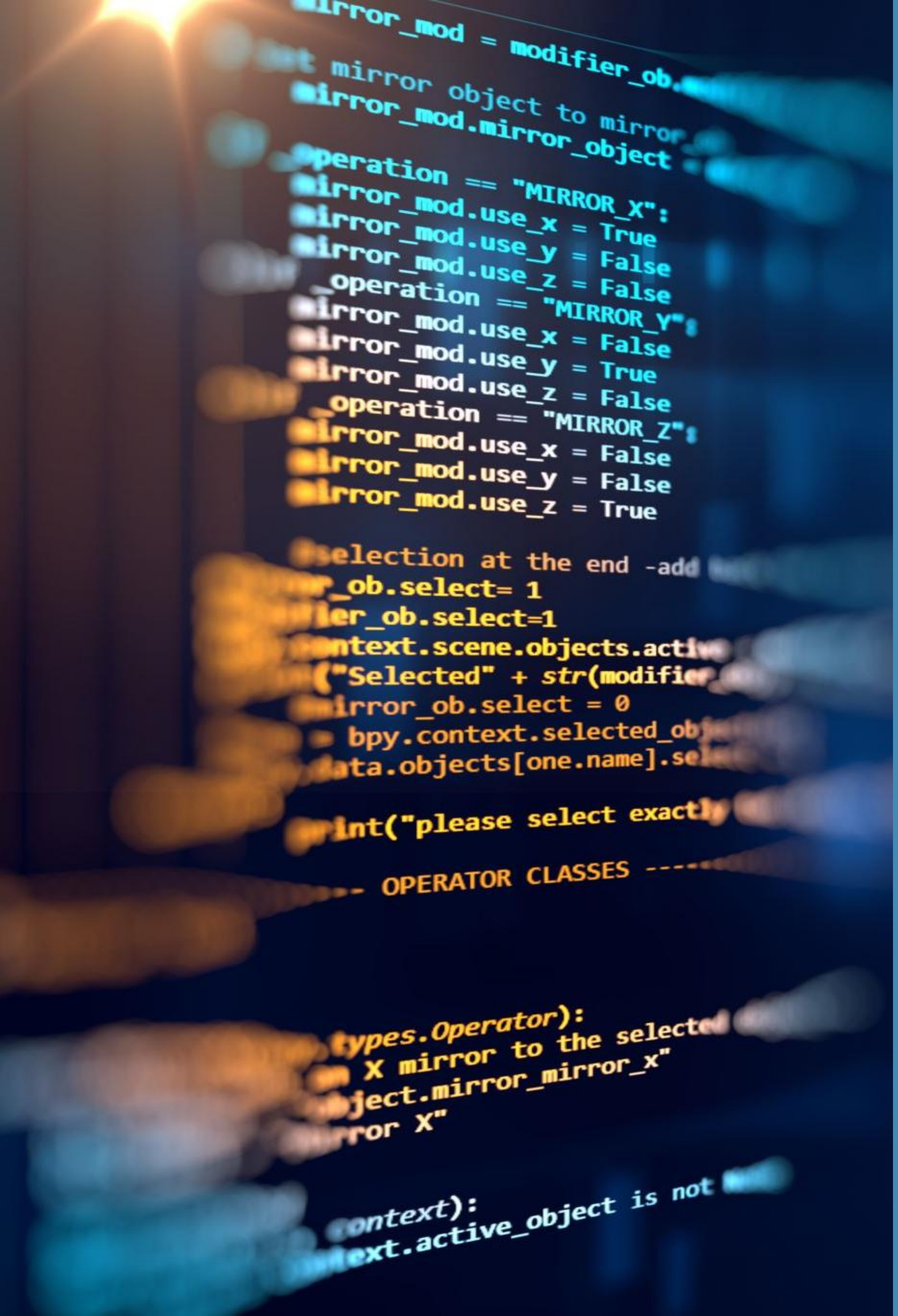
OUR FINAL PROJECT: BUILD A PHYSICAL COMPUTING "INTER-FACE"

MTEC1005 | Physical Computing | Fall 2023

Sean Michael Landers

Final Project

LET’S INTRODUCE THE FINAL PROJECT!



```
irror_mod = modifier_ob.
 mirror object to mirror
irror_mod.mirror_object =
operation == "MIRROR_X":
irror_mod.use_x = True
irror_mod.use_y = False
irror_mod.use_z = False
 operation == "MIRROR_Y":
irror_mod.use_x = False
irror_mod.use_y = True
irror_mod.use_z = False
 operation == "MIRROR_Z":
irror_mod.use_x = False
irror_mod.use_y = False
irror_mod.use_z = True

selection at the end -add
_ob.select= 1
ier_ob.select=1
ntext.scene.objects.active
("Selected" + str(modifier_
irror_ob.select = 0
 bpy.context.selected_obj
ata.objects[one.name].se

rint("please select exactly

--- OPERATOR CLASSES ----

types.Operator):
 X mirror to the selected
ject.mirror_mirror_x"
rror X"

context):
xt.active_object is not
```

FINAL PROJECT MISSION:

PROTOTYPING A PHYSICAL COMPUTING INTER-FACE

PROTOTYPING A PHYSICAL COMPUTING INTER-FACE?

01

PROTOTYPING a physical computing inter-face

02

Prototyping a **PHYSICAL COMPUTING** inter-face

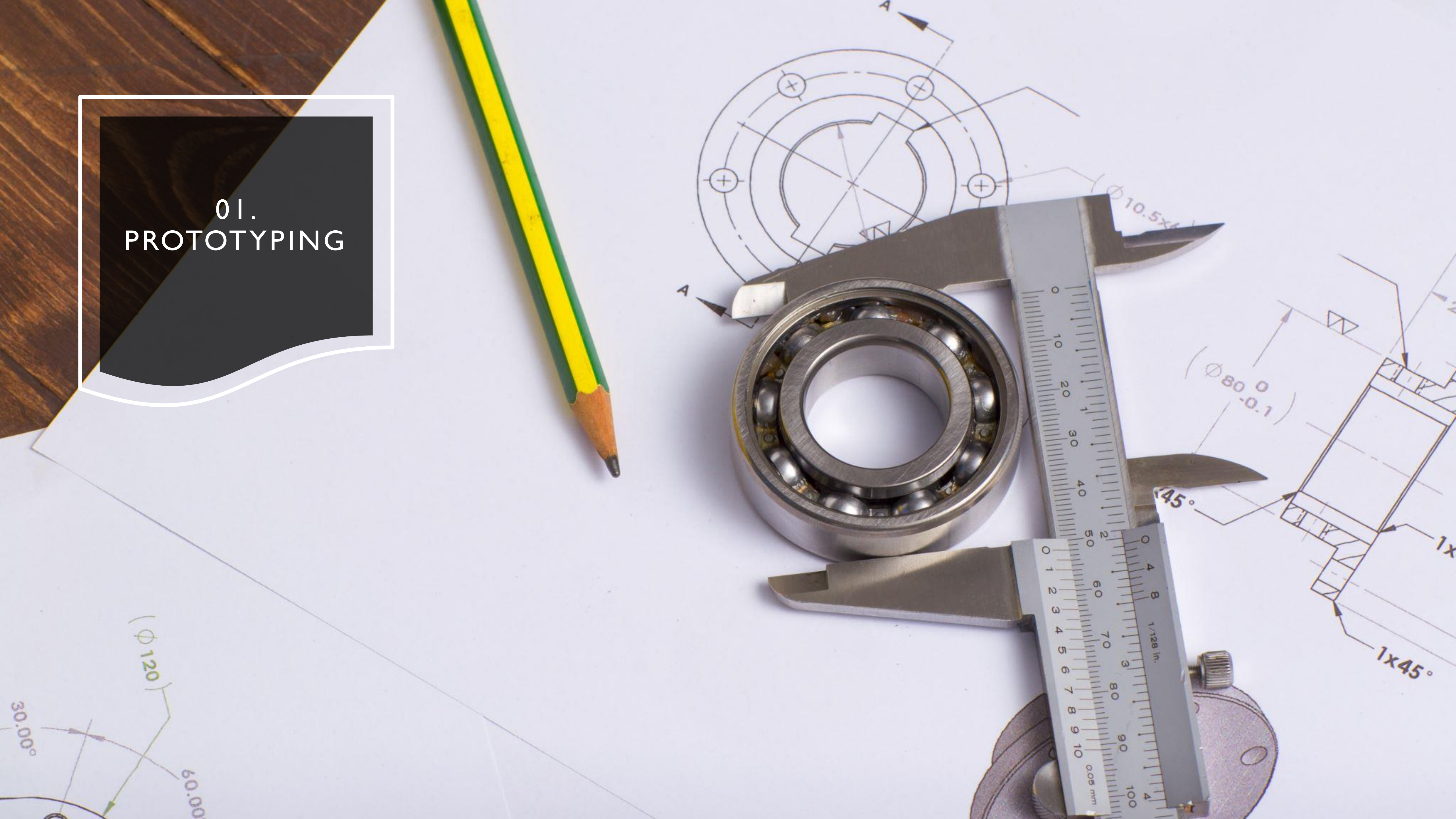
03

Prototyping a physical computing **INTER-FACE**

04

Requirements for prototyping a physical computing interface

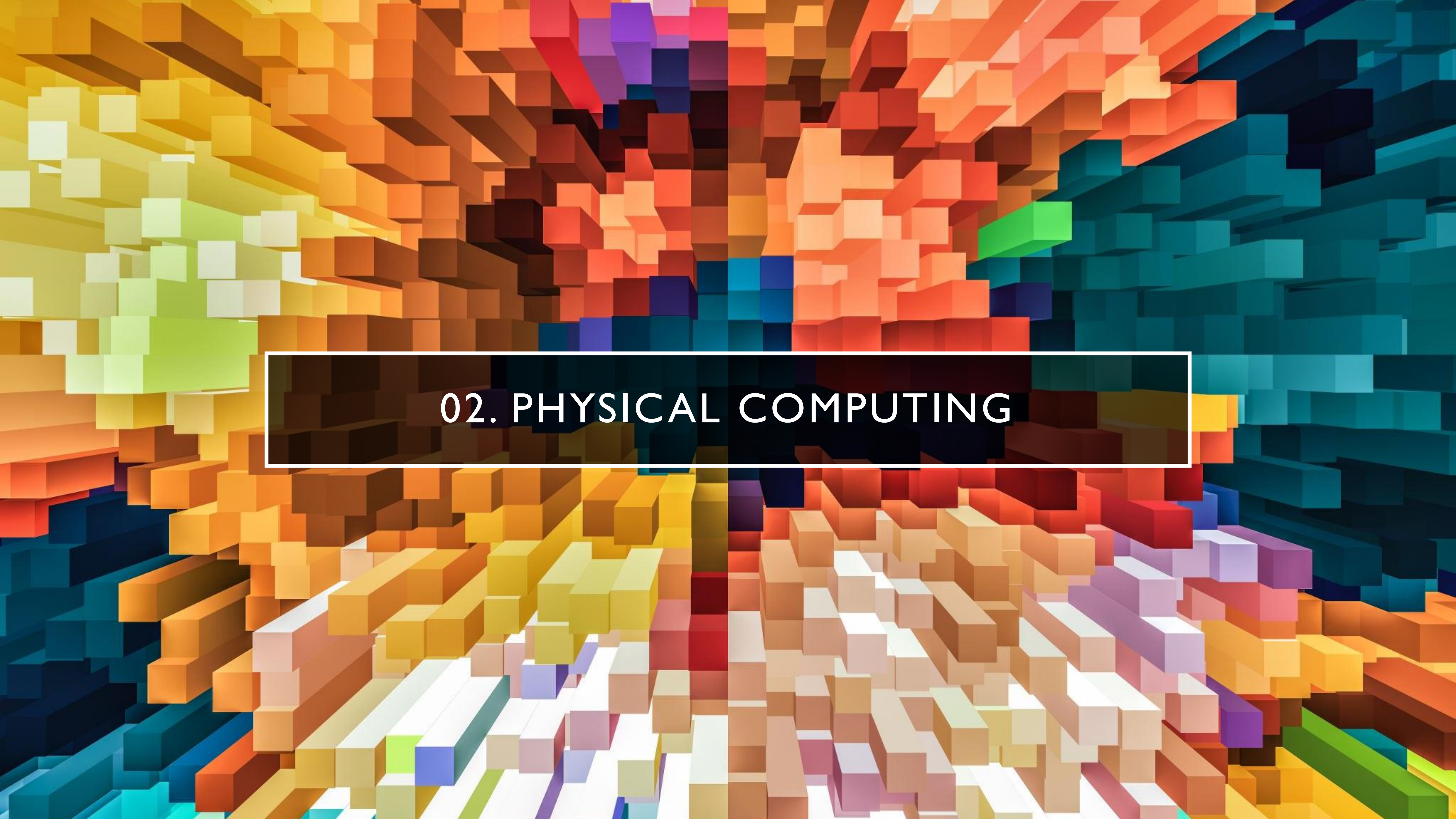
PROTOTYPING A PHYSICAL COMPUTING INTER-FACE?



01. PROTOTYPING

Building a **proof of concept** model to **test** out an idea

PROTOTYPING



02. PHYSICAL COMPUTING

Sensing and controlling **the physical world** through hardware and software

PHYSICAL COMPUTING

PHYSICAL COMPUTING

Connecting electronic and physical media through (mostly) **human-centered interactive systems**

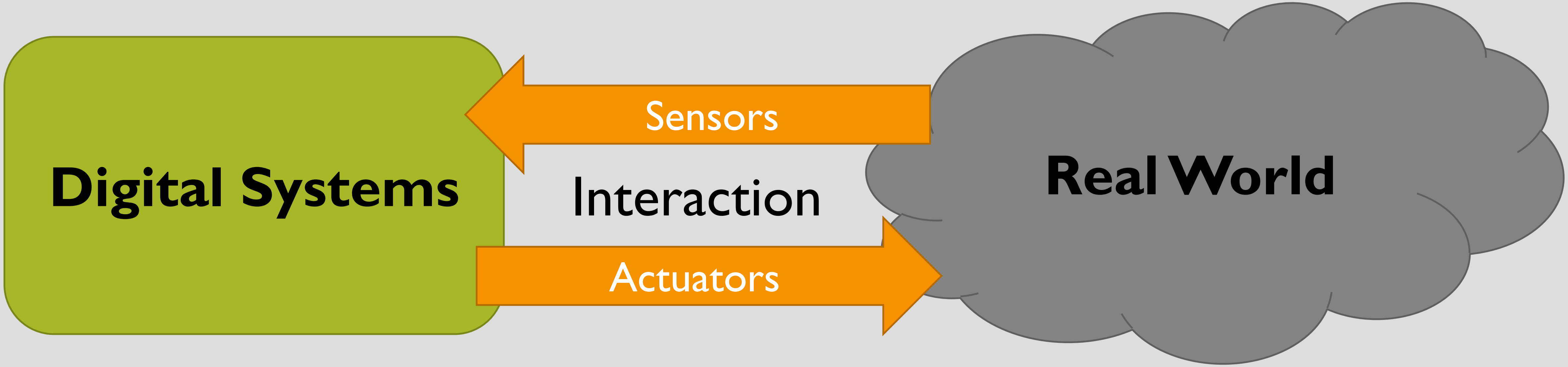
Augmenting physical objects and environments with the ability to **sense, see, talk, move, generate, connect, and "be smart"**

PHYSICAL COMPUTING

Prioritizes **body-centered** computing interfaces

PHYSICAL COMPUTING

PHYSICAL COMPUTING



This design is a rework of Nevit Dilmen's 2014 CC0 1.0 work "Physical Computing."

03. INTER-FACE

INTERFACE

“A place of **interaction between two systems.**”

-Marshall McLuhan, *Understanding Media:The Extensions of Man* (1962)

A means for **communication** between **human** and **computing systems**

INTERFACE

INTER –

Prefix, from the Latin, "between"

INTER-FACE

– FACE

Appearance

Facial Expression

Surface

04. PROJECT REQUIREMENTS

PROJECT REQUIREMENTS

Prototype a physical computing “inter-face”

The interface must have at minimum:

- A physical “face” of some kind (as abstract or as concrete as you’d like to make it, literal or figurative)
- One of these **sensors**:

  Push button, potentiometer, LDR/photocell
- One of these **actuators**:

  Servo motor or piezo
- At least two LEDs

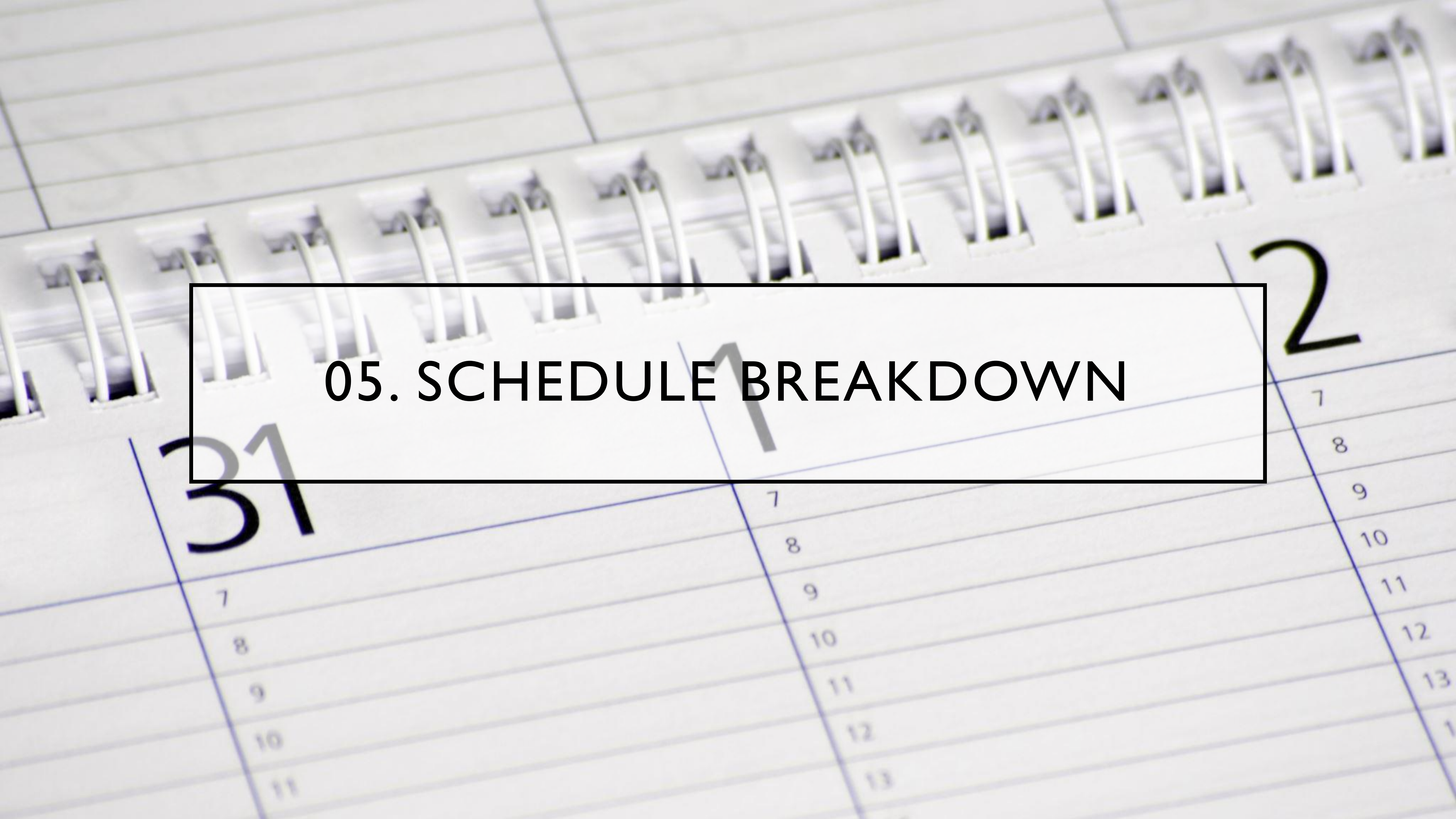
PROJECT REQUIREMENTS

The final inter-face must:

- Be **reactive or interactive** (with functioning **inputs** and **outputs**)
- Be **presentable and testable** (ie: not falling part)
- Have **physically embedded components within the "face"**

PROJECT REQUIREMENTS

This is a **solo** project
BUT you are ENCOURAGED to
work with others
for troubleshooting
and developing ideas further.



05. SCHEDULE BREAKDOWN

SCHEDULE

Week 11:

PROJECT BRAINSTORMING

Week 12:

SYSTEM PROTOTYPING + FAB PLANNING

Week 13:

FABRICATION + INTEGRATION (LO-FI PROTOTYPING)

Week 14:

PLAYTESTING AND ITERATION

Week 15:

EXHIBITION AND FINAL DELIVERABLES DUE

WEEK 11 DELIVERABLES

SKETCH / DIAGRAM OF PROJECT IDEA

LIST OF PROJECT COMPONENTS

BRIEF SUMMARY OF INTERACTION

(SEND VIA SLACK BEFORE NEXT CLASS)

WEEK 12 DELIVERABLES

CODE IN PROGRESS – INTEGRATING ALL INPUTS/OUTPUTS

FABRICATION MATERIALS LIST

REVISION OF WEEK 11 DELIVERABLES AS NECESSARY

(SEND VIA SLACK BEFORE NEXT CLASS)

WEEK 13 DELIVERABLES

PHOTO DOCUMENTATION OF FABRICATION IN PROGRESS

UPDATES TO CODE, IF APPLICABLE

(SEND VIA SLACK BEFORE NEXT CLASS)

WEEK 14 DELIVERABLES

PHOTO DOCUMENTATION OF UPDATES, IF APPLICABLE

UPDATES TO CODE, IF APPLICABLE

(SEND VIA SLACK BEFORE NEXT CLASS)

WEEK 15 DELIVERABLES

IN-CLASS PRESENTATION OF FINAL PROJECT

VIA SLACK:

FINAL PHOTO AND VIDEO DOCUMENTATION

FINAL CODE

06. FOR YOUR REFERENCE…

FOR MORE INFO ON TESTING CIRCUIT CONNECTIONS…

For reference:

http://arduinotogo.com/2016/08/22/chapter-2-using-the-multimeter/

FOR MORE INFO ON TESTING VOLTAGE…

For reference, see:

http://arduinotogo.com/2016/10/07/chapter-4-voltage/

FOR MORE INFO ON TESTING CURRENT…

For reference, see:

http://arduinotogo.com/2016/10/07/chapter-4-current/

FOR MORE INFO ON TESTING RESISTANCE…

For reference, see:

http://arduinotogo.com/2016/10/07/chapter-4-resistance/

https://arduinotogo.com/2017/03/10/appendix-a-reading-resistor-codes/

Voltage, Current, Resistance

	VOLTAGE	CURRENT	RESISTANCE
LED			
RESISTOR			
BATTERY			

SOURCE: HTTP://ARDUINOTOGO.COM/2016/10/07/CHAPTER-4-REVIEW/

Voltage, Current, Resistance

	VOLTAGE	CURRENT	RESISTANCE
LED	The LED will get dimmer as the voltage gets lower, or brighter as more voltage is added; if there is too much voltage the LED will burn out	LEDs only need a very small amount of current to run. However, reducing the amount of current too much will turn the LED off.	LEDs have a tiny amount of resistance
RESISTOR	Voltage is converted into heat when it crosses over a resistor. More voltage means more heat and less voltage means less heat.	Resistors lower the amount of current being drawn in a circuit,	The amount of resistance depends on the resistor's rated value. Check appendix b to learn how to identify resistor values
BATTERY	Batteries establish the voltage level for both the high point and zero volts, a.k.a. the ground.	Current comes from the battery. The current flowing will change depending on what components attached to the battery and how much current they require.	Since a battery is not a perfect conductor, there is a small amount of resistance inside of the battery, but when it is in our circuits it is effectively zero.

SOURCE: HTTP://ARDUINOTOGO.COM/2016/10/07/CHAPTER-4-REVIEW/

VOLTAGE, CURRENT, RESISTANCE

Ohm's Law:

V (voltage) = I (current) * X (resistance)

OTHER WAYS OF EXPRESSING OHM'S LAW

$V = I * R$

$I = V / R$

$R = V / I$

FOR MORE INFO ON OHM'S LAW…

For reference, see:

http://arduinotogo.com/2016/10/07/chapter-4-ohms-law/

FOR MORE INFO ON COMPONENTS IN PARALLEL / SERIES

For reference, see:

http://arduinotogo.com/2016/10/07/chapter-4-components-in-parallel-and-series/

Arduino Code Review

Structure

```
/* Each Arduino sketch must contain the
following two functions. */

void setup()
{
 /* this code runs once at the beginning of
the code execution. */
}

void loop()
{
 /* this code runs repeatedly over and over as
long as the board is powered. */
}
```

Comments

```
// this is a single line

/* this is
a multiline */
```

Setup

```
pinMode(pin, [INPUT \ OUTPUT]);
/* Sets the mode of the digital I/O pin.*/
```

Control Structures

```
if(condition){
  // if condition is TRUE, do something here }
 else {
// otherwise, do this
}
```

Digital I/O

```
digitalWrite(pin, val);
/* val = HIGH or LOW write a HIGH or a LOW value to a digital
pin. */

int var = digitalRead(pin);
/* Reads the value from a specified digital pin,
either HIGH or LOW. */
```

Arduino Code Review

Data Types

```
void  //nothing is returned
boolean  //0, 1, false, true
int  //16 bit integer (i.e. -5, 400, 32)
float // 32 bit decimal (i.e. -5.2, 77.835)
```

Constants

```
HIGH \ LOW
INPUT \ OUTPUT
true \ false
```

Mathematical Operators

```
= //  assignment
+ //  addition
- //  subtraction
* //  multiplication
/ //  division
```

Logical Operators

```
== //  boolean equal to
!= //  not equal to
< //  less than
> //  greater than
<= //  less than or equal to
>= //  greater than or equal to
&& //  Boolean AND
|| //  Boolean OR
! //  Boolean NOT
```

More at https://www.arduino.cc/reference/

SEE YOU NEXT TIME!

IMAGE CREDITS

Slide 9. "Voltage" by sparkfun is licensed under CC BY 2.0
Slide 10. "Current" by sparkfun is licensed under CC BY 2.0
Slide 11. "Resitance" by sparkfun is licensed under CC BY 2.0
Slide 14-16 are slides designed by Josh Corn (and modified by Sean M. Landers), and licensed under CC BY-NC-SA 4.0