Lecture 10
Processing and Arduino topics

IAT267 Introduction to Technological Systems

Topics for today

- Controlling servomotors with Arduino
- More code examples
- Networking - intro

Organizational items

- Quizes: one more quiz in Unit 11
  - This is an optional quiz and it will replace one of your lowest marks in the other quizzes
- Quiz of week 9 –we are aware of the exiting issues with one of the questions and will fix all grading errors manually over this weekend
- Project milestone 3 due November, 16

Workshop

- Multiple sensors exercise from last week
  - If you have not finished this exercise in class – work on it on your own
  - Arduino boards, sensors, breadboard available from the Library

- This week – no workshops
  - Remembrance Day

Controlling Servomotors with Arduino

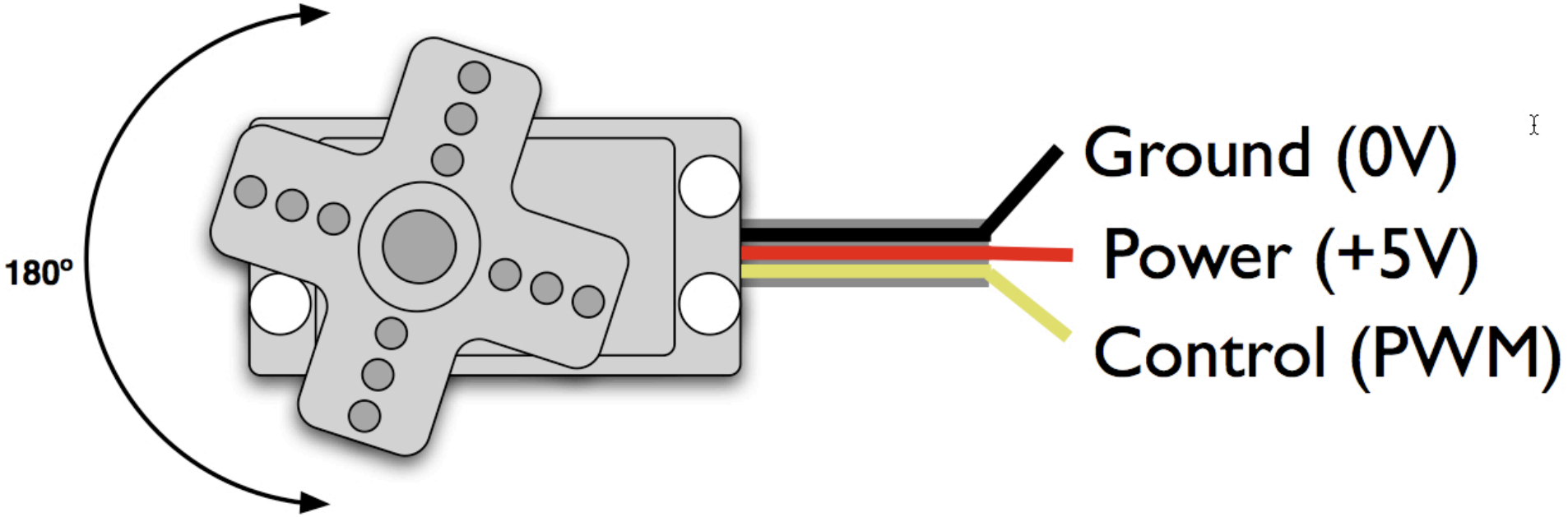
Servomotors

- Can be positioned from 0-180º (usually)
- Easy three-wire PWM, 5V interface

Uses of servomotors

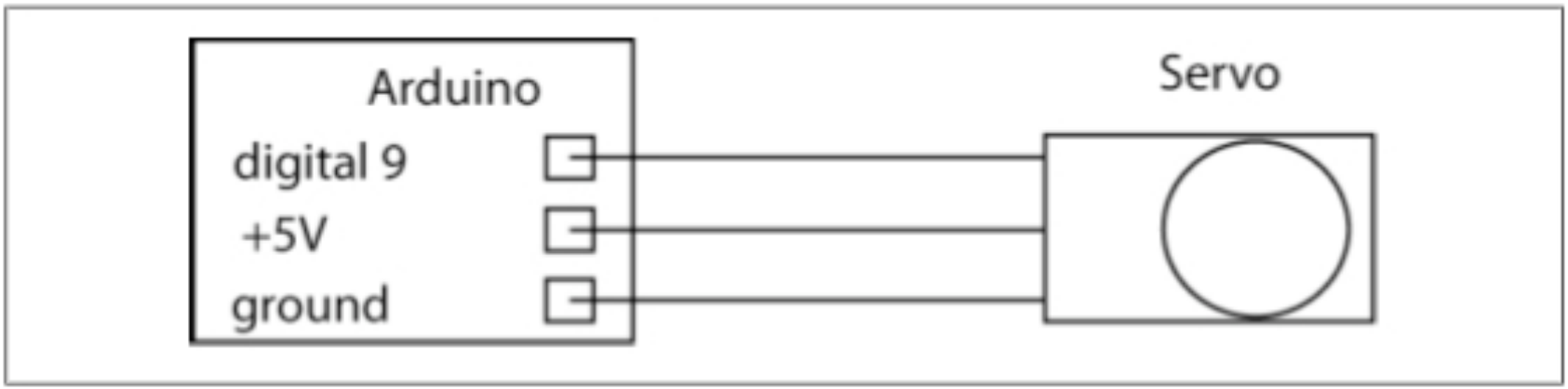
- Many objects that you want to turn or rotate can be manipulated with a servo:
  - Cameras
  - Mirrors
  - Directed lightning, etc
- An easy correlation between a potentiometer and a servo can be established: turn the potentiometer and the servo turns

Servo Control

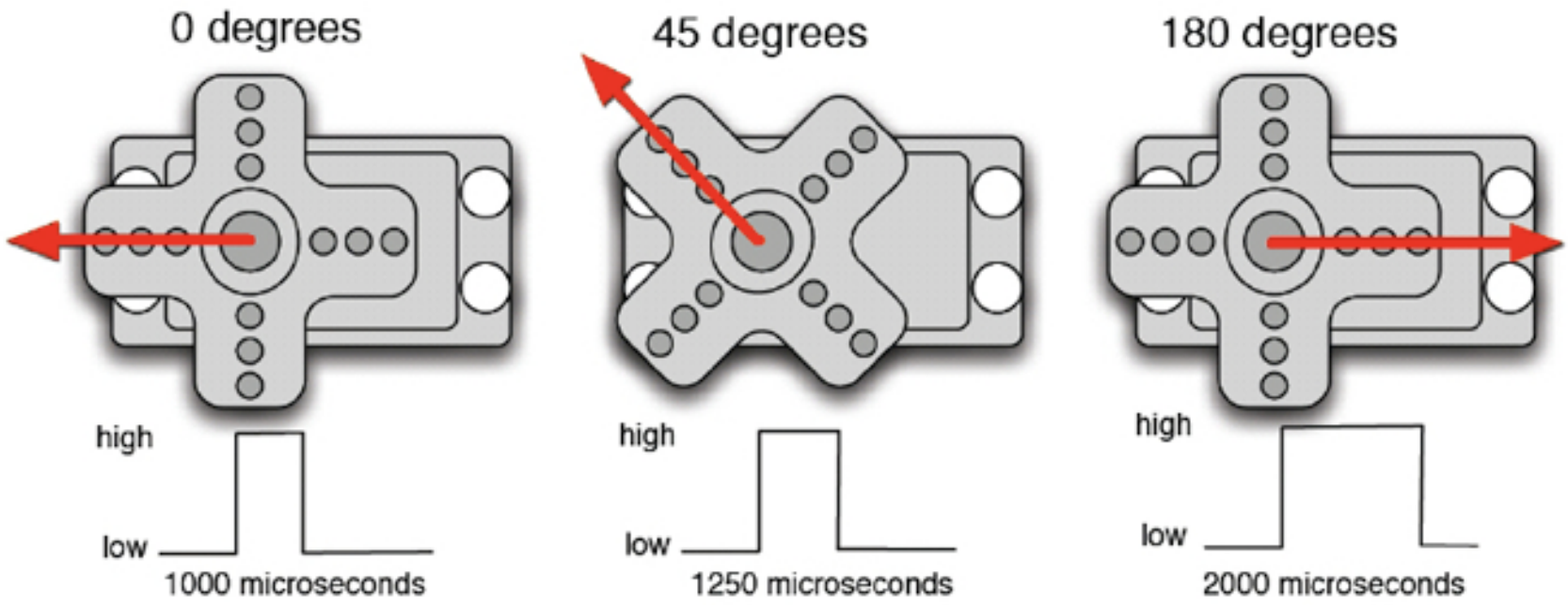


Servo connection to Arduino

- Need to provide the servomotor with power from the 5V of the Arduino board
- Ground connection – to the GND on the Arduino board /breadboard
- Example of connection on the next slide


Arduino
digital 9
+5V
ground
Servo

Angles of Rotation:



code

- Controlling a servo is just a matter of determining how long in microseconds you need to send each command to the servo
  - Writing a pulse to the motor that is the desired length
- Example: a value between 0 and 180 degrees (followed by a non-digit character) is sent using the serial connection to a computer to the servo
- See the servo spec sheet

Arduino Servo Library

- This library is included with the Arduino IDE
- It lets Arduino control one or more servomotors
- Arduino servo library: http://www.arduino.cc/en/Reference/Servo
- Three methods:
  - attach()
  - write()
  - read()

attach() method

- Starts reading the Servo instance on the pin passed in to the method

servo.attach(pin);

servo.attach(pin, min, max);

//optionally set the min and max pulse widths

write() method

- Writes a value to the servo.
- On a standard servo, this will set the angle of the shaft (in degrees)
- On a continuous rotation servo, this will set the speed of the servo
  - 0 = full speed in one direction
  - 180 = full speed in the other direction
  - 90 = no movement

read() method

- Reads the current angle of the servo (the value passed to the last call to write()).

Sample code

- On webct:
  - Servo1.pde: using a potentiometer to control position of servo
  - Servo2.pde: serial control
  - RemoteServo.pde – control of multiple servomotors

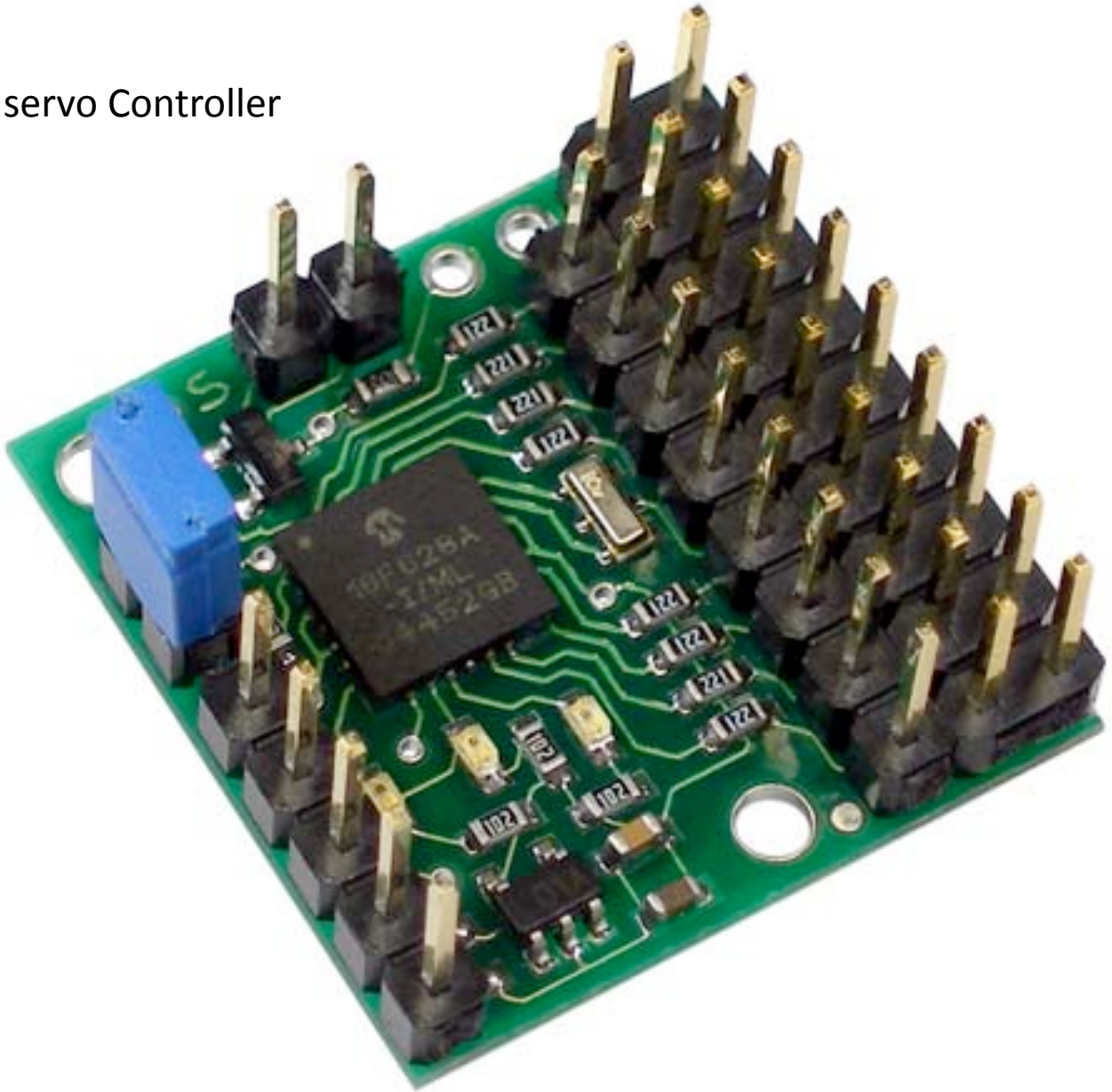
    - The Servo Library on Arduino supports up to 12 servos (48 servos for the Arduino Mega).

If you need more servos

- Servos draw considerable power, so if you need more than one or two, you will need to power them up from a separate supply
- Connect the grounds of the Arduino and external power supply together

- If you need more servos than Arduino can handle (more than 12) – you can use a dedicated servo controller device (see example on next slide)

Pololu Micro Serial servo Controller



http://www.pololu.com/catalog/product/207

Advantages of using servomotors

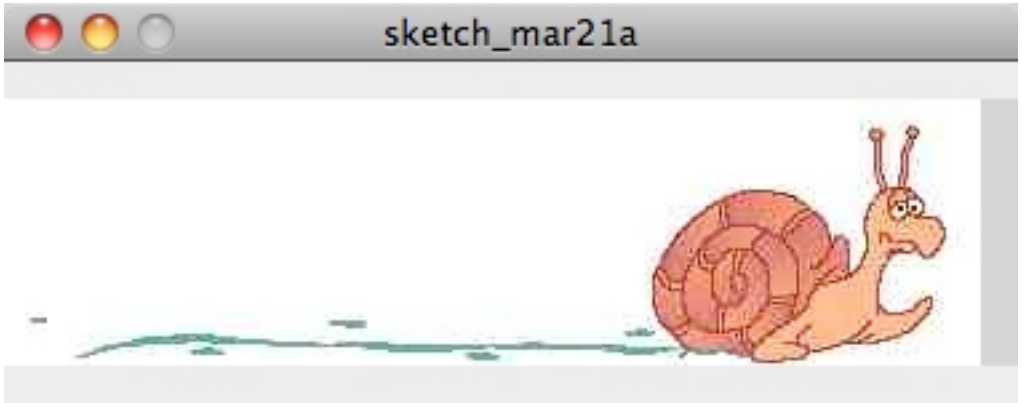
- Easy to control + precision
- A servo reads the amount of voltage passed in to determine how far to rotate within a limited range
  - The most common range for a servo is 180 degrees
  - The servo can move from any point in that range to any other point clock - / counterclock wise

Advantages of using servomotors

- Powerful tools for controlling anything that needs a controllable, repeatable movement
- There are many to choose from:
  - Different sizes and performance
- The signals used to control servos are almost always the same

Animating a sequence of images in Processing

- You need first to add all the images to your sketch folder
- See code posted on webct



Networking
Protocols and Communication

- Communicating over networks
- Using XML
- Understanding networks and the internet
- Handling network communication in Processing

Data in an application

- User: gets data from an application:
  - To understands what the application is doing
  - How the user actions have been interpreted
  - How to get the application to do what the user wants

  →FEEDBACK

- Data sent by the user to the application

  →INPUT

Data exchange

- So far we have focused on data exchanged between the user and application (input or feedback)

- In this unit, we will focus on *exchanging data between an application and devices*
  - this is not new: we already have seen Processing application communicating with devices connected to Arduino

Data exchange

- We will look at the factors that influence communication between **different machines and applications:**
  - Context
  - The means of communication
  - Machines used
  - Amount of data being exchanged

Protocols

- Are set of rules governing communication between devices
  - Cell phones
  - Communication over the internet
  - Devices plugged into your computer
- Enabling communication between devices:
  - Means ensuring that all devices know the same protocol (‘talking the same language’)

Protocols

- What is being communicated
- How that communication is going to be encoded
- What channels the communication is going to take place on
- Speed of transmission

Benefits of communicating over networks

- Enable your applications to work remotely
- Get information from remote locations
- Send information to other devices, applications, and locations
- Send commands to a remote server
- Gather data from the internet
- Network multiple devices and machines

User Interaction

- The interaction in a design does not always need to be a matter of the user interacting with the interface of your application

- In most interactive application: a great part of the communication goes on between the application and other devices or applications

- Data conversion: for example, if data collected from sensors is to be transmitted over a network, it will have to be converted in a format according to the protocol being used.

Communicating over networks

- Remote data
- Remote access
- Remote control
  - Using these opens up a wide range of possibilities in designing applications
  - Some examples in next slides

Sample apps: Telectroscope

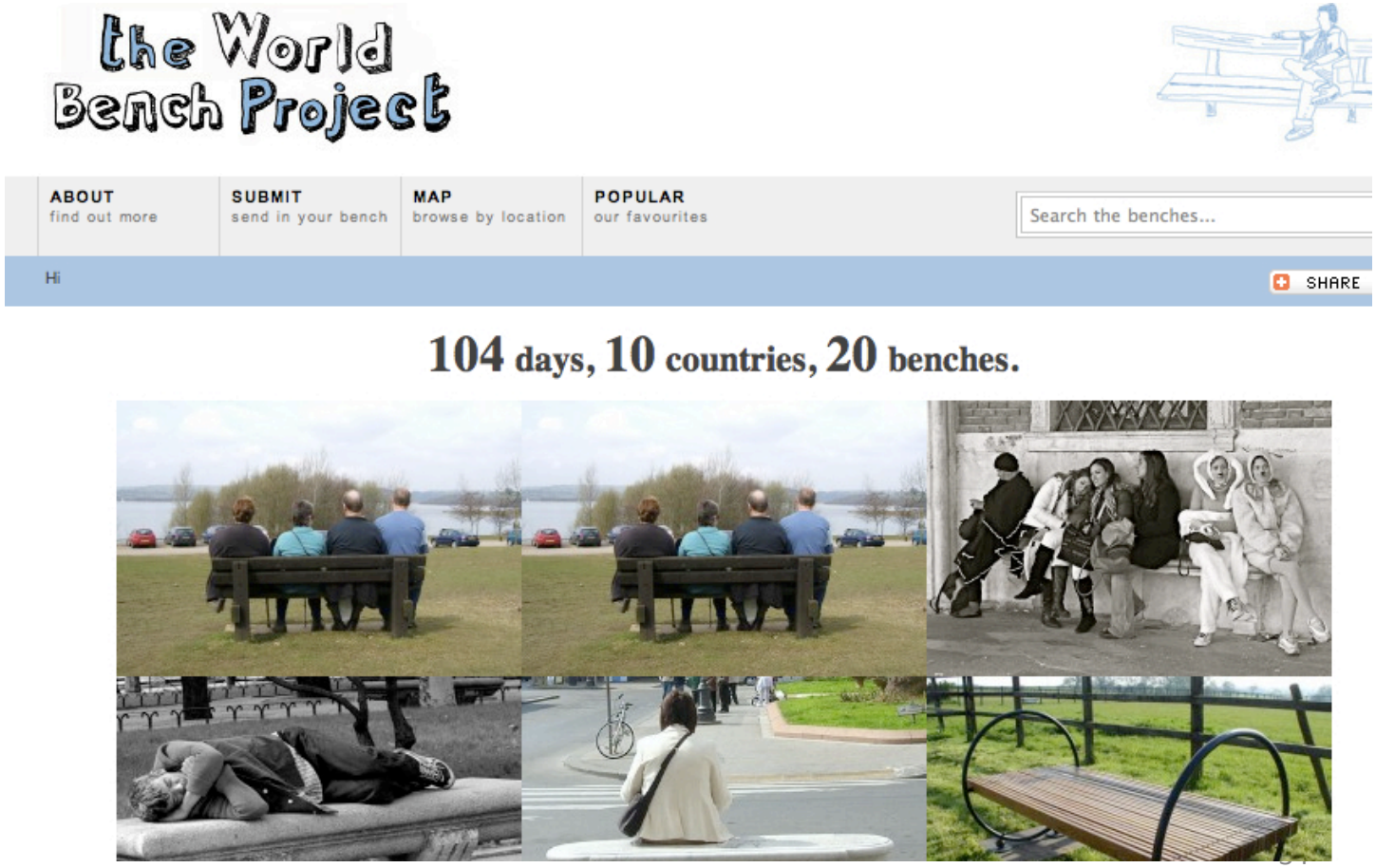
http://www.talktalk.co.uk/telectroscope

- Send a video feed across the Atlantic Ocean
- Provide a video link between London and New York City

Sample app: World Bench

http://www.theworldbenchproject.com/

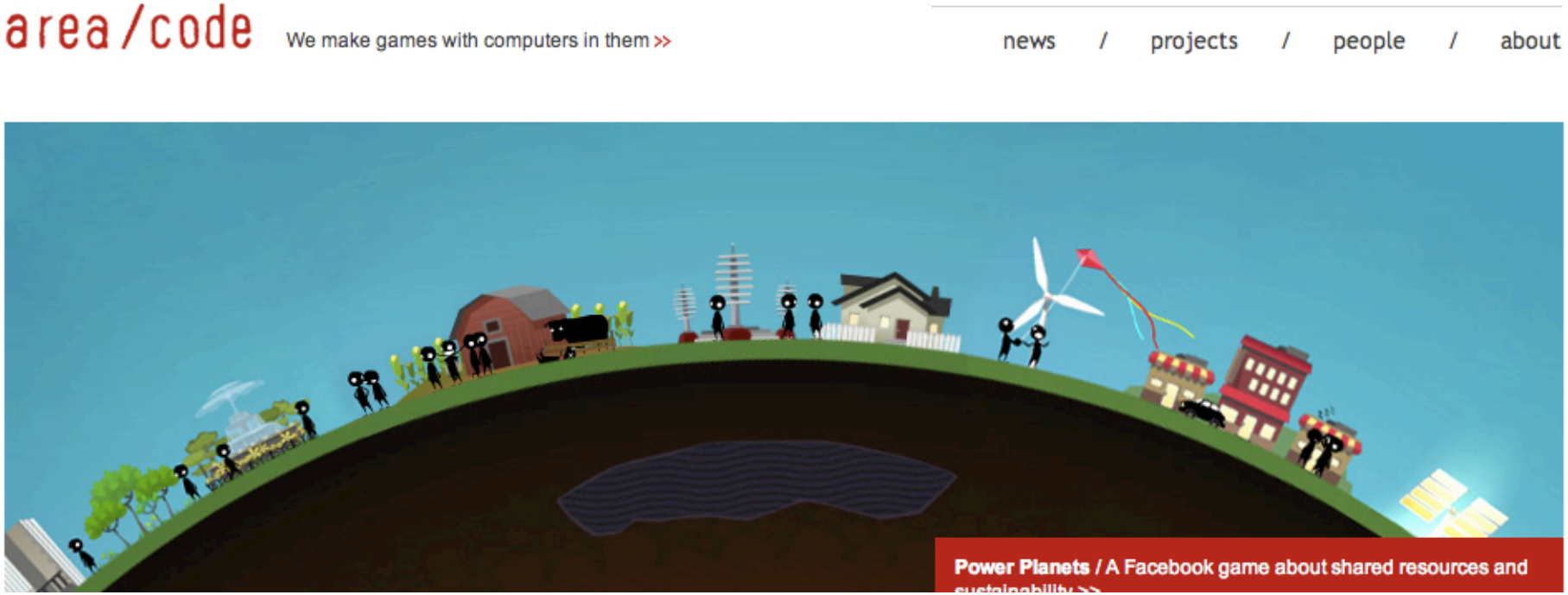
- Mirror public benches in locations across the world



Sample app: Area/ Code

http://areacodeinc.com/

- Area/Code games highlight the connections between the interactive systems and imaginary landscapes inside of games and the real world around them.



Sample app: Tele-Garden

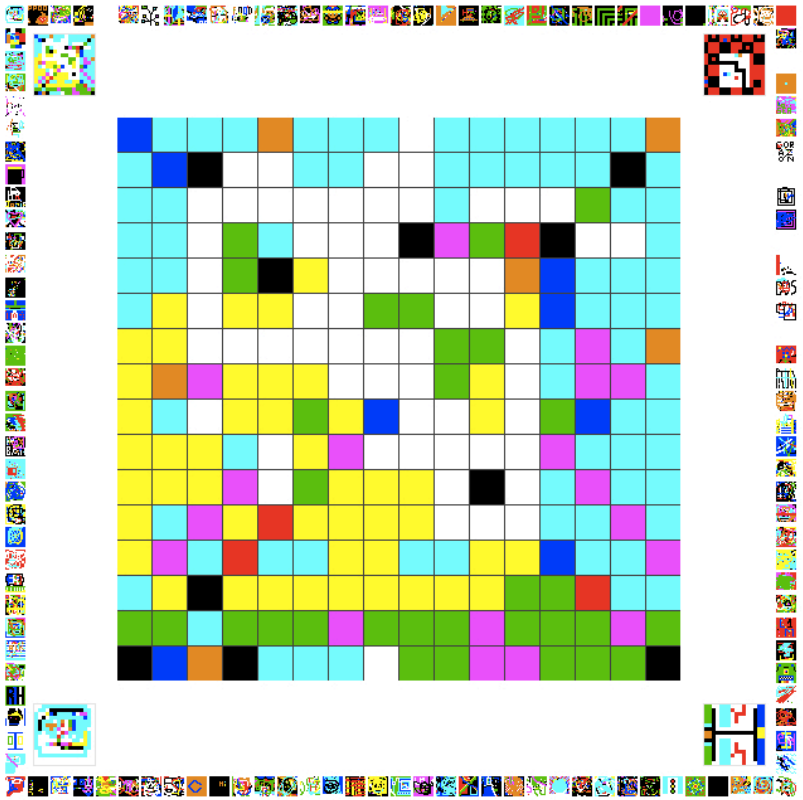
http://www.usc.edu/dept/garden/

- An installation of a garden that is controlled remotely by thousands of remote gardeners from all over the globe who operate robotic arms that arrange light, water and plants.

Sample app: Screening Circle

http://artcontext.net/screeningCircle/

- Creates shared space where multiple users can create images together in a sort of sewing-circle collaborative effort.



- Networks: uses:
  - Cover greater amounts of space between parts of an application by spreading an application over multiple machines
  - Gather data from another source
  - Allow users who are not physically present
  - Allow users to connect with one another
  - Pass processing off to more powerful computers

Adding networking capability to an application

- Is much easier with tools that are provided in Processing and Arduino
- The main challenge of networking and networks is making them efficient and fast
  - Depends on the scale of the project

Using XML

- Extensible Markup Language (XML) is a way of giving a document structure.
  - Very commonly used by Internet web services – Twitter, RSS feeds
  - A very common way to store and send data
  - Sometimes, when we want to get data from another source, the data is in XML format

Data

- Things represented as data have a few common characteristics
  - Specific **traits**
  - They **have** things that belong to them
  - They **belong** to other things

Example: a library

- **Characteristics**: street address, name
- **Has**: books, magazines, DVDs
- **Belongs-to**: library system, street, city

Example of XML code

```
<library>
    <book>
        <title>Great Expectations</title>
        <author>Charles Dickens</author>
        <publication_year>1883</publication_year>
    </book>
</library>
```

- The library **has** a book
  - The book **has** an author and a title

- The idea of ownership is expressed by nesting the nodes within one another.

  - Node = a single object within the document that has some associated data

Processing and XML

- Processing has an XML library to make creating and parsing XML easier
  - XMLElement class
    - Contains methods for:
      - loading XML files,
      - creating new XML files,
      - looking through data for specific pieces of information,
      - setting new values in an existing XML file or dataset

Sample Code

- addressTOcoord code
  - See sample code on webct

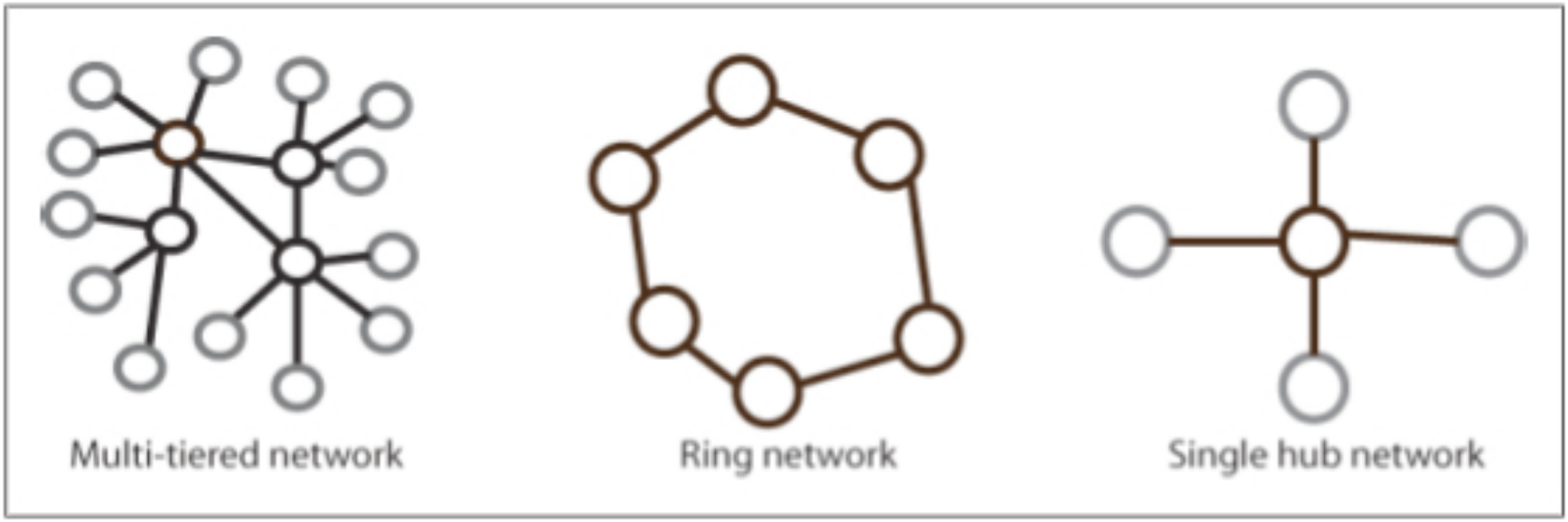
- Google Geocoding API
  - http://code.google.com/apis/maps/documentation/geocoding/

Understanding networks and the internet

- All networks have a few things in common:
  - Machines on the network:
    - Need to be identifiable to each other and themselves
    - Need to know how to connect with one another
    - Need to know what protocol the other machines are using

Network Organization

- There a few different ways to organize a network



Hub network

- Similar to connecting multiple devices to Arduino

- Other examples:
  - Bluetooth-enabled networks
  - A group of computers connected to a single central server that sends commands out to each of the nodes on the network

Ring network

- Any message has two possible paths
  - This also means, in a worst-case scenario, that to get a message from one machine to another, you might have to send n/2 messages where n is the number of machines in the network
- Example: a group of Arduino controllers connected together

Multi-tiered network

- Is just a collection of single-hub networks that have ways to determine which of the central nodes is acting as a hub to the desired child node

- This is how the Internet is set up

Communication modes

- Two network processes can communicate as:
  - Client-server
  - Peer-to-peer

Client-Server Communication

- With client/server communication, communication is asymmetrical, and the server program is different from the client program.
- The server provides a service to the client. It is usually the client who initiates the communication or transaction.
- A server can be communicating to multiple clients at the same time.

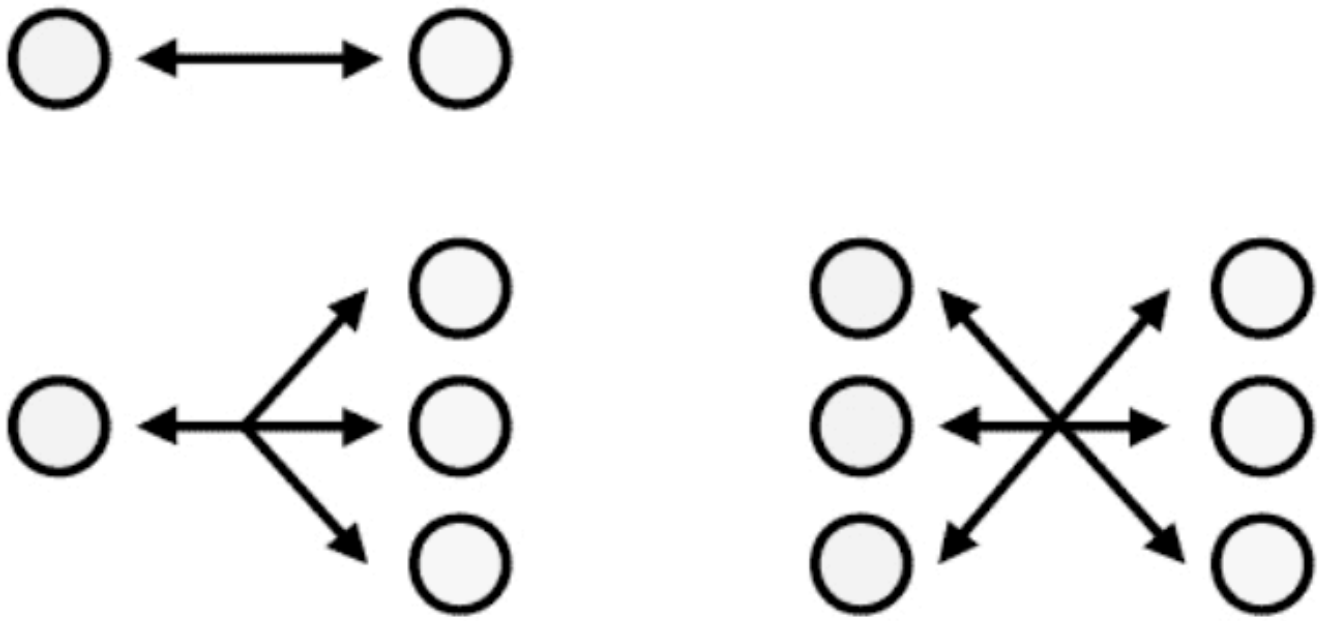
Peer-to-Peer Communications

- With peer-to-peer communication, communication is symmetrical, and the two communication processes are based on the same application program. Either end can initiate communication or transaction.

- Examples of network programs that operate in peer-to-peer mode include Microsoft NetMeeting and some chat programs.

- Note that one peer can be communicating to multiple peers at the same time.

Communication Patterns (1)



Communication Patterns (2)

- **one-to-one**: two processes are involved in one communication session, exchanging messages between each other, for example, Internet phone

- **one-to-many or many-to-one**: in a communication session, one process exchanges messages to and from a group of processes, for example, remote lecturing

- **many-to-many**: messages are exchanged among members in a group of communication processes, for example, video-conferencing

Resources

- For the code and examples on the slides:
  - **Processing: a programming handbook for visual designers and artists**/ Casey Reas, Ben Fry/ The MIT Press/ Cambridge, Massachusetts, London, England (this book is available in the Library)
  - **Processing - Creative Coding and Computational Art** / Ira Greenberg
  - Code examples: www.processing.org

Thank you!

Questions?