Below is a simple sketch that supports wired PEER-to-PEER serial communication between UNOs using the single hardware UART (Rx/Tx) for local Serial Monitor support and a software serial support on pins 7 and 8. (*Use with the ACES Serial Communication Stick may require using a different pair of software serial support pins*: [http://darcy.rsgc.on.ca/ACES/PCBs/index.html#UARTSPI](http://darcy.rsgc.on.ca/ACES/PCBs/index.html%23UARTSPI))

**// PROJECT  :Chat**

// PURPOSE  :Utility to enable a peer-peer conversation between ACES' Arduino users

// DEVICE   :2 Arduinos

// AUTHOR   :Many hands

// DATE     :2020 02 15. Updated 2025 02 13.

// uC       :328p

// COURSE   :ICS3U-E

// STATUS   :Working

// REFERENCE:https://www.arduino.cc/en/Tutorial/SoftwareSerialExample

//          :http://darcy.rsgc.on.ca/ACES/Projects/SerialMastermind/index.html

// NOTES    :Hardware serial used to Rx/Tx from/to local Serial Monitor

//          :Software serial used to Tx/Rx to/from peer

#include <**SoftwareSerial**.h>

**SoftwareSerial** chat(7,8); // RX, TX

#define BAUD 9600         //Common BAUD rate for hardware UART and Software Serial

char ch;                  //single utility character for communication

void setup() {

**Serial**.begin(BAUD);     //use the hardware UART for the Serial Monitor

 while (!**Serial**);        //wait for serial port to connect

**Serial**.println("Starting Chat Program...");

 chat.begin(BAUD);       //set the data rate for the SoftwareSerial port

 delay(1000);                        //let things settle...

 chat.println("Hello, world?");      //anyone home?

**Serial**.println("Ready to chat..."); //local connection ready...

}

void loop() {

 if (chat.available()) {             //anything incoming?

//if so, display the oldest character in the incoming (queued) buffer

**Serial**.write(chat.read());

  }

 if (**Serial**.available())  {          //anything outgoing?

**Serial**.print("Self:\t");          //if so, let's get started...

   while (**Serial**.available()) {      //transmit each of the outgoing characters...

     ch = **Serial**.read();             //...to the software serial stream

     chat.write(ch);                 //...

//also, echo each character on my local Serial Monitor

**Serial**.write(ch);

}

//at end, transmit a newline (carriage return) character

    chat.println();

//at end, also write a newline character to the local Serial Monitor

**Serial**.println();

  }

}

### Chat Prototype

