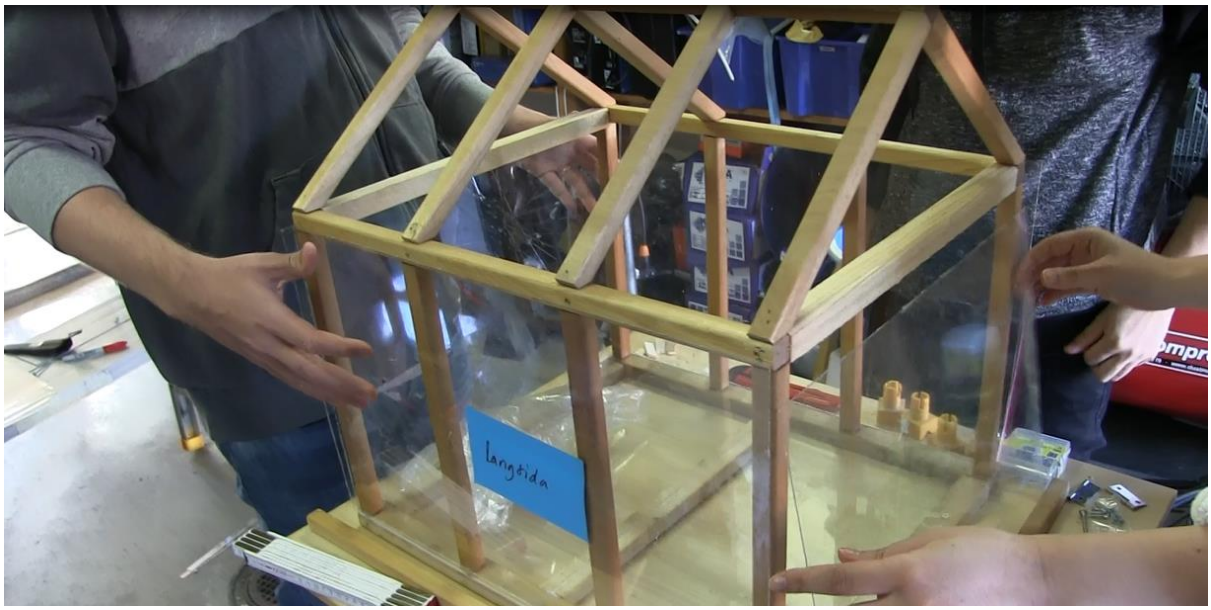
SUPERGUIDE FÖR AUTOMATISERAT VÄXTHUS



Innehållsförteckning

Introduktion

Automatiserat växthus

Ett växthus fungerar ungefär som jordens atmosfär. Stora delar av solens energi strömmar igenom glasrutorna och värmer upp luften i växthuset. Glasrutorna hindrar sedan den uppvärmda luften från att strömma ut ur växthuset, detta gör att luften i huset blir mycket varmare än luften utanför. På det viset kan man odla olika typer av växter som annars inte klarar av det kallare klimatet genom att reglera luftfuktighet och temperatur som passar dem.

Att underhålla ett växthus och odla växter som kräver en viss temperatur, fuktighet och/eller vattenmängd i odlingen kräver både tid och energi. Tänk om det fanns ett sätt att låta allting regleras automatiskt?

Det är här ett automatiserat växthus kommer in. Ett automatiserat växthus består bland annat av olika sensorer som mäter de viktigaste faktorerna i en odling, exempelvis temperatur. Med hjälp av programmering går det sedan att göra så att en lucka i taket öppnas och stängs med hjälp av en motor baserat på temperaturen. Du kan även starta vattenpumpen om det skulle bli för torrt i växthuset.

Följ denna manual för att bygga ditt alldeles egna automatiserade växthus!

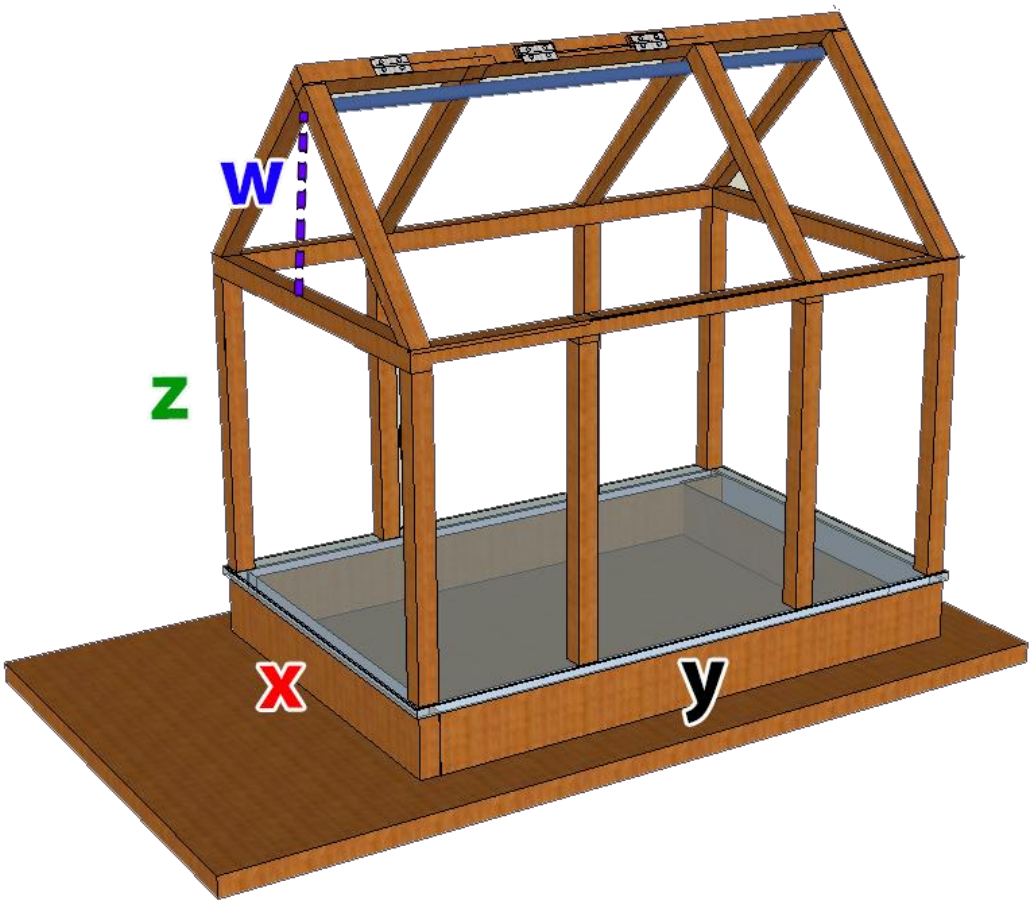
Framställning av växthuset

Nedan följer alla nödvändiga delar som möjliggör framställningen av växthuset. Läs igenom hela manualen noga i förväg och följ sedan stegen i ordningen de är skrivna i. Detta är endast ett förslag på hur man kan bygga växthuset, hitta gärna på egna!

1. **Planera**
   Tänk noga igenom alla steg innan ni börjar att inhandla material och bygga växthuset. Om ni känner att ni har frågor inom ett visst område så tveka inte att höra av er till personer med mer kunskap, dem flesta svarar mer än gärna på frågor.

2. **Material och verktyg**

Basen:	2-3 st Xcm träreglar kortsida, 2st Ycm träreglar långsida, 1st X*Y plexiglas
Väggarna:	8st Z cm träreglar, 2st Z*Y cm plexiglas (långsidorna), 2st Z*X cm plexiglas (kortsidorna)
Taket:	8st $\sqrt{W^2 + \left(\frac{x}{2}\right)^2}$ cm träreglar, 1st Ycm träregel, 1st vattenslang, Y cm Qst gångjärn
Övrigt:	Xst skruvar, 1st vattenbehållare, 1st teknikbehållare



3. **Bygga växthuset**
   Det finns massor av sätt att bygga huset på. I denna manual kommer det finnas en grundlig genomgång av ett växthus med bevattning i taket. Det är fullt möjligt att göra det på andra sätt också. Vi ger ett ytterligare ett förslag på utförande där växthuset har bevattning direkt i krukorna . Egna idéer är också välkomna!

Växthus med bevattning i taket

1. Lägg upp basen först så ni har någonting att bygga på.

2. Ta fyra stycken träreglar; två för kortsidan och två för långsidan, och placera dem på träskivan. Basen kommer vara "golvet" på huset och träskivan kommer vara till hjälp när huset flyttas runt.

3. Placera plexiglaset, som ska vara botten, på basen.

4. Mät och lägg ut trä-reglarna som ska vara stöd till väggarna.

5. Skruva fast träskivan med botten-plexiglaset och trä-reglarna nerifrån och upp. Se till att skruvarna är tillräckligt långa för att skruva ihop materialet samt en marginal för att få det stabilt. **OBS: Börja alltid med hörnen och jobba dig inåt.**

6. Skruva ihop kortsidorna med botten-plexiglaset till träbotten. Det här ska skruvas uppifrån och ner. **OBS: Man kan fästa reglarna på olika sätt.**

7. Dags att bygga taket! Ta fyra nya träreglar som är lika långa som de i botten och placera ut långsidorna över de stående, redan tillsatta, väggreglarna. Ta fram två mindre träreglar på vardera sida av huset som ska bilda husgaveln.

8. Fäst ihop reglarna.Tänk på att lämna plats i taket för att lägga in växterna. **OBS: Var försiktig så att reglarna ej spricker.**

9. Fäst ihop takets kortsidor med lämplig fästningsmetod.

10. Skär ut plexiglas enligt husets mått som ska vara fyra väggar och flera mindre skivor som ska vara tak. Det är viktigt att glasen täcker hela huset så att det blir en bra isolering.

11. Sätt fast plexiglasen. Tänk på att minst två av de mindre tak-delarna ska fästas även med gångjärn för att det ska kunna ventileras.

12. Fäst bevattningsslangen i taket och gör små hål som vattnet kan rinna ut ur. Detta kan göras förslagsvis med krokar med större böj på.

Växthus med bevattning direkt i krukorna

1. Steg 1-11 ovan är identiska för dessa två designen.

2. Istället för att fästa slangen i taket lägger man den direkt i krukan. Om man har flera krukor kan man dela upp slangen i flera mindre slangar.

4. Koppla på tekniken

Innan vi börjar med tekniken behöver vi inhandla lite material. Det som behövs är:

- Arduino (Om det finns med inbyggt wifi köp det, det förenklar senare steg.)
- Vattenpump (12V som mest)
- Temperatur och fuktighetssensor (exempelvis en DHT-22)
- (Valfritt) Wifi modul ESP8266 (endast om arduinon inte har inbyggt wifi), denna används till att skicka data till internet, exempelvis Thingspeak
- Resistorer (Köp ett paket med flera olika motstånd)

- Transistorer (PN2222)
- Dioder (1N4001)
- Kablar (Olika typer är bra, Hane till Hane, Hane till Hona, Hona till Hona)
- Servo(5V)
- Fläkt (12v som mest)
- Strömadapter med minst 1A och 12V
- Brödbräda

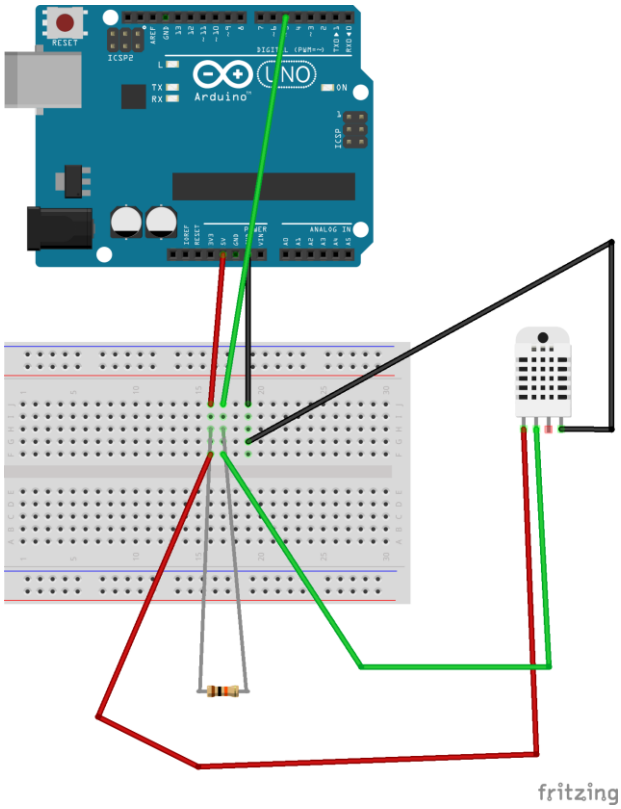
För att koppla upp all tekniken följ kopplingsscheman noga och dubbelkolla dem innan du startar något program.

Fuktighet- och temperatursensor DHT-22

Denna sensor ska användas för att mäta temperaturen och fuktigheten i växthuset. Med hjälp av dessa värden kommer vi kontrollera servot och fläkten.

Koppla upp DHT22 enheten

Denna sensor kopplas enligt kopplingsschemat nedan. Notera att pinne 3 från DHT-22sensorn inte används.
Ground kopplas till den högra pinnen sett framifrån.
Styrningspinen, som är den andra från vänster, skall kopplas till någon utav I/O portarna 1-13. På exemplet nedan har vi kopplat den i nummer 5.
Den vänstra pinnen kopplas till 3.3 eller 5 V.
Mellan Strömmen och styrningspinnen skall det sitta en 10K resistor enligt kopplingsschemat.



Pröva DHT22 enheten

Börja med att ladda ned biblioteket för att kunna använda DHT-sensorn. Denna hittar ni på deras github klicka på clone&download och sedan väljer ni zip : https://github.com/adafruit/DHT-sensor-library. För att lägga till biblioteket i Arduino klickar ni på sketch i er IDE och sedan väljer ni "Include Library". Därefter väljer ni "Add .ZIP Library" och väljer filen ni precis laddade hem.

Nu är vi redo att börja skriva koden! Skapa en ny sketch i Arduino IDE och börja med att ladda in biblioteket och definiera de olika pinnarna samt typen av DHT-sensor, vi skapar även de olika variablerna som kommer att behövas..

```
#include "DHT.h"
#define DHTPIN 13     // Digital pin that the sensor is connected to
#define DHTTYPE DHT22   // DHT 22
DHT dht(DHTPIN, DHTTYPE);
//Read values from sensors
float fHumidity;
int iHumidity;
float fTemp;
int iTemp;
```

I setup metoden vill vi initiera dht sensorn.

```
void setup() {
	//Starting serial so we can print temp and humidity
	Serial.begin(9600);
	dht.begin();
}
```

Nu vill vi skapa en metod för att läsa värdena från sensorn.

```
void readSensor() {
	// Reading temperature or humidity takes about 250 milliseconds!
	// Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
	fHumidity = dht.readHumidity(); //Reads data from sensor
	// Read temperature as Celsius
	fTemp = dht.readTemperature();
	 // Check if any reads failed and exit early (to try again).
	if (isnan(fHumidity) || isnan(fTemp) ) {
		Serial.println("Failed to read from DHT sensor!");
	}
}
```

Nu vill vi köra denna metoden i loopen varannan sekund.

```
void loop() {
	readSensor(); //Reads the values from the sensor
	//Prints the values to the serial monitor
	Serial.print("Humidity: ");
	Serial.print(fHumidity);
	Serial.println(" %\t");
	Serial.print("Temperature: ");
	Serial.print(fTemp);
	Serial.println(" *C ");
	delay(2000); //2 seconds until new value
}
```

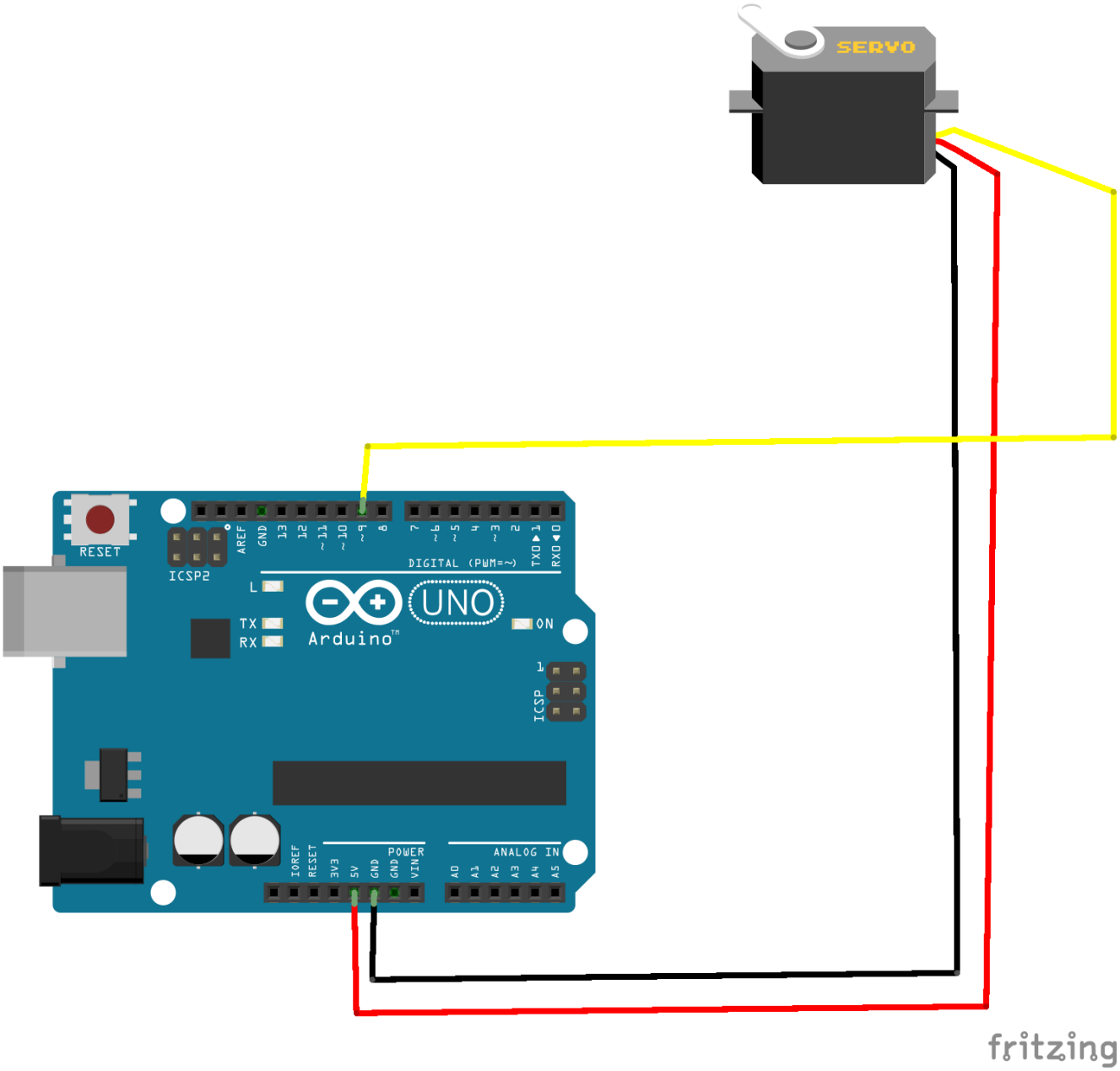
Kontrollera koden genom att trycka på verify i Arduino-programmet. Kontrollera även kopplingarna en gång innan uppladdningen av koden till Arduino-enheten. För att se mätvärdena som skrivs ut trycker ni på "Tools" och sedan på "Serial Monitor". Detta är ett bra sätt för att kontrollera att allt står rätt till! Den fullständiga koden till DHT22 sensorn hittar ni på hemsidan.

Servo

Servot kommer användas till att öppna och stänga fönstret. Detta görs med hjälp av en arm som lyfter upp en lucka och på så vis släpper ut varm luft från växthuset.

Koppla upp servon

Servon kopplas enligt schemat nedan.
Servot har tre kablar; Strömkabeln är ofta röd och skall kopplas till 5V pinnen på arduinon, Ground-kabeln är ofta svart eller brun och skall kopplas till ground pinnen på Arduinon, Signal-kabeln är oftast gul eller orange och skall kopplas till en av I/O-pinnarna 1-13 på Arduinon.



Pröva servon

För att kunna använda oss utav servot behöver vi inkludera ett bibliotek för denna och skapa en variabel för servot. Vi vill även skapa variabler för hur många grader det skall vara när fönstret är öppet och stängt. Vi skapar också en variabel för att veta om fönstret är öppet och variabler för servopinnen och dess positionen.

```
#include <Servo.h>
Servo myservo;

//Servo variables and constants
const int windowDegreeOpen = 120;
const int windowDegreeClosed = 70;
int servoPosition = 0;   // variable to store the servo position
boolean windowOpen = false; // To check if window is closed or open
```

Nu vill vi skapa en metod för att få servot att gå till en viss position.

```
/**
  This method turns the servo
*/
void topos(int gotopos) {
        myservo.attach(9); // Makes the servo ready for use
        delay(3000);
        if (servoPosition <= gotopos) {
                for (servoPosition; servoPosition <= gotopos; servoPosition += 1) { // goes
                from 0 degrees to 180 degrees
                        // in steps of 1 degree
                        myservo.write(servoPosition);// tell servo to go to position in variable
                        'pos'
                        delay(100); // Adjusts the speed of the servo when it moves into
                position
                }

         }
        else
        {
                for (servoPosition; servoPosition >= gotopos; servoPosition -= 1) { // goes
                from 0 degrees to 180 degrees
                        // in steps of 1 degree
                                myservo.write(servoPosition); // tell servo to go to position in
                        variable 'pos'
                                delay(100);
                        }
                }
        myservo1.detach(); //Detaches servo so it won't respond to current
}
```

Nu vill vi i loopen prova att flytta servon.

```
void loop() {
	//Moves the servo then waits to seconds and moves it back
	topos(windowDegreeOpen);
	delay(2000);
	topos(windowDegreeClosed);
	delay(2000);
}
```

För att förenkla till senare skapar vi två metoder som sköter fönsteröppningen åt oss.

```
void openWindow(){
	if(!windowOpen){
		topos(windowDegreeOpen);
		windowOpen = true;
	}
}

void closeWindow(){
	if( windowOpen){
		topos(windowDegreeClosed);
		windowOpen = false;
	}
}
```

Nu kan vi enkelt skicka in true för att öppna fönstret och false för att stänga det. Det kan se ut såhär.

```
void loop() {
	openWindow();
	delay(2000);
	closeWindow();
	delay(2000);
}
```

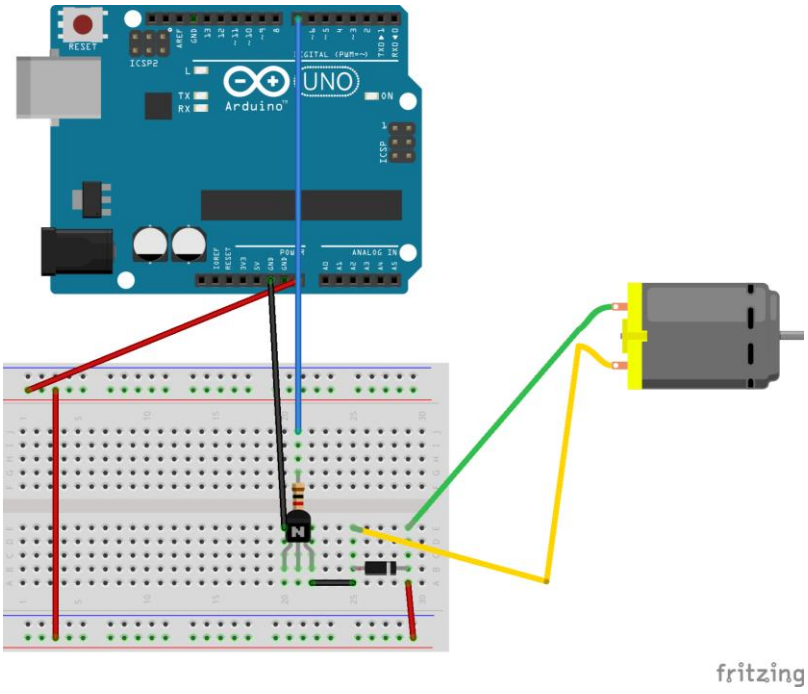
Kontrollera koden genom att trycka på verify i Arduino-programmet. Kontrollera även kopplingarna en gång innan uppladdningen av koden till Arduino-enheten. Nu kan vi prova att köra programmet och se om servon rör sig. Den fullständiga koden till servon hittar ni på hemsidan.

Pumpen

Pumpen kommer att användas för bevattning av växterna i växthuset. Pumpen vi har valt är en peristaltisk pump vilket innebär att den inte tar skada även om den körs utan vatten.

Koppla upp pumpen

1. Kopplas enligt kopplingsschemat nedan. Här är det extra viktigt att koppla rätt för att skydda arduinon.
2. Pin 7 på arduinon skall kopplas till mittenbenet på transistorn. Se bilden nedan. Denna kopplas med en 1K resistor emellan.
3. Ground kopplas till det vänstra benet på transistorn. Sett med den platta sidan mot er.
4. Det högra benet på transistorn kommer till slut att gå till den negativa sidan på motorn. Men den skall först kopplas med en diod mellan den och strömmen enligt kopplingsschemat nedan.
5. Strömmen kopplas till Vin porten på arduinon som kommer att ge ut 12 V när strömadaptern är ikopplad. Den kopplas sedan genom samma spår som dioden innan den kopplas till plussidan på motorn.
6. Dioden är till för att försäkra att strömmen inte ska åka tillbaka in i arduinon och förstöra den.
7. Transistorn är till för att skydda arduinon från höga volttal.



Pröva pumpen

Först definieras vilken pin som pumpen är kopplad till på arduinon och den initieras sedan i setupmetoden, vi definierar även en variabel för när pumpen användes senast.

```
const int pumpPin = 6;
unsigned long lastWateredMillis;
```

```
void setup() {
        pinMode(pumpPin, OUTPUT);
}
```

Nu vill vi skapa en metod för att starta pumpen och en för att stoppa den.

```
void startPump() {
        digitalWrite(pumpPin, HIGH);
}
void stopPump() {
        digitalWrite(pumpPin, LOW);
        lastWateredMillis = millis();
}
```

Nu kan vi i loop metoden prova att starta och stoppa pumpen.

```
void loop() {
        startPump();
        delay(5000);
        stopPump();
        delay(2000);
}
```

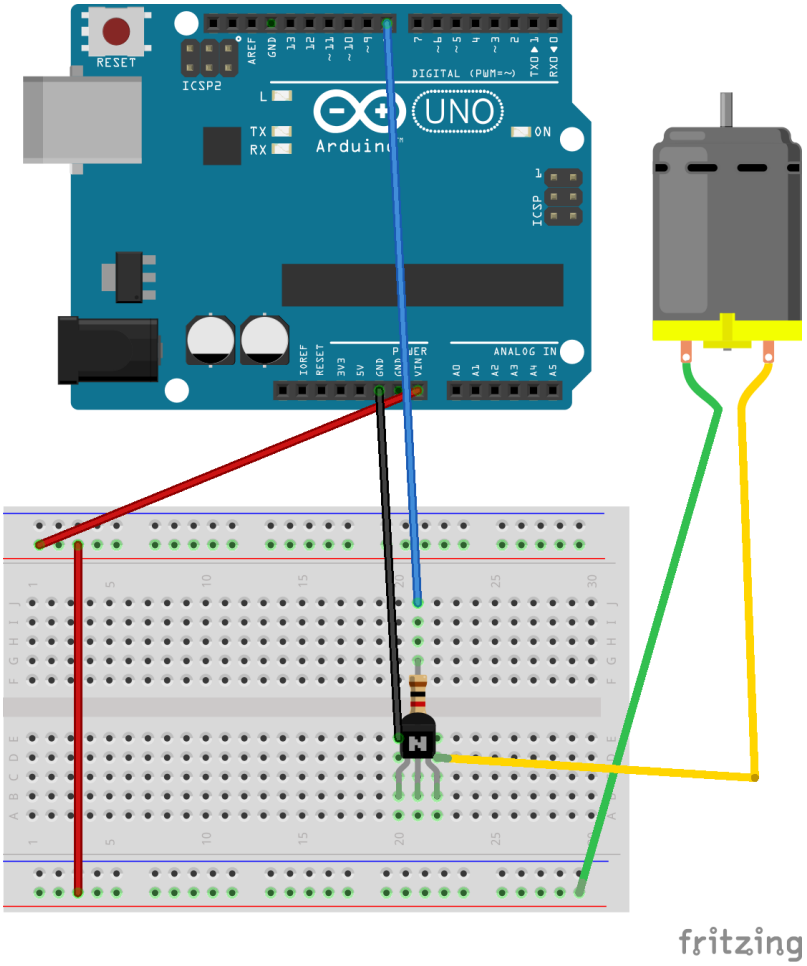
Kontrollera koden genom att trycka på verify i Arduino-programmet. Kontrollera även kopplingarna en gång innan uppladdningen av koden till Arduino-enheten. Nu kan vi prova att köra programmet och se om pumpen börjar pumpa. Den fullständiga koden till pumpen hittas ni på hemsidan.

Fläkten

Fläkten kommer att användas för att cirkulera luften i växthuset vid hög temperatur och fuktighet.

Koppla upp fläkten

1. Fläkten kopplas enligt schemat nedan. Här är det precis som med pumpen extra viktigt att koppla rätt för att skydda arduinon.
2. Pin 8 på arduinon skall kopplas till mittenbenet på transistorn. Se bilden nedan. Denna skall kopplas med en 1K resistor.
3. Ground kopplas till det vänstra benet på transistorn. sett med den platta sidan mot er
4. Det högra benet på transistorn skall gå till den negativa sidan på motorn. Transistorn är till för att skydda arduinon från höga volttal.
5. Strömmen kopplas till Vin porten på arduinon. Den kopplas sedan till fläkten.



Pröva fläkten

Först definieras vilken pin som fläkten är kopplad till på arduinon och den initieras sedan i setupmetoden.

```
const int fanPin = 8;
boolean fanStatus = false;

void setup() {
	pinMode(fanPin, OUTPUT);
}
```

Därefter skapar vi metoder för att starta och stoppa fläkten.

```
void startFan() {
	if(!fanStatus){
		analogWrite(fanPin, 250);
		fanStatus = true;
	}
}
void stopFan() {
	if(fanStatus){
		digitalWrite(fanPin, LOW);
		fanStatus = false;
	}
}
```

I loop-metoden så vill vi prova så att allt funkar genom att starta och stoppa fläkten med 5 sekunders mellanrum.

```
void loop() {
	startFan();
	delay(5000);
	stopFan();
	delay(2000);
}
```

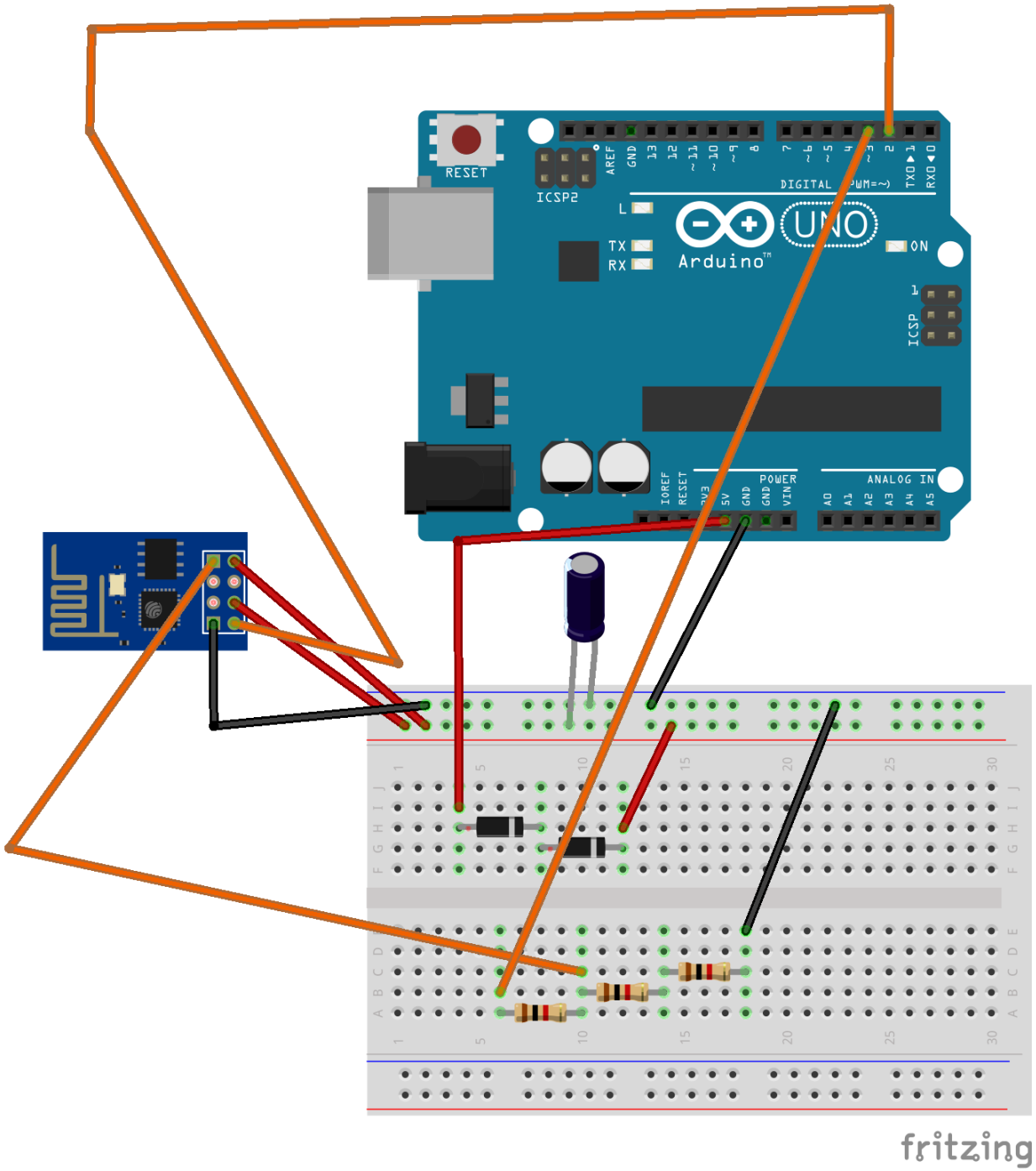
Kontrollera koden genom att trycka på verify i Arduino-programmet. Kontrollera även kopplingarna en gång innan uppladdningen av koden till Arduino-enheten. Nu kan vi prova att köra programmet och se om fläkten startas. Den fullständiga koden till fläkten hittar ni på hemsidan.

Wi-Fi modul

Med hjälp av Wi-Fi modulen kommer man att kunna skicka data till en hemsida som heter thingspeak. Denna hemsida kan visa en graf över all data som skickas till den, tex temperaturen och fuktigheten i växthuset. För att det ska fungera behövs ett Wi-Fi nätverk som modulen kan koppla upp sig mot. För att uppkopplingen mellan Wi-Fi modulen och nätverket ska fungera behöver man veta nätverkets namn och lösenord. För att skicka upp data till thingspeak behöver man skapa ett konto och hämta något som kallas för en API-nyckel.

Koppla upp Wi-Fi modulen

1. Wi-Fi modulen kopplas enligt kopplingsschemat nedan.
2. Port 2 på arduinon skall kopplas till TX porten på wifi modulen. Detta syns på kopplingsschemat nedan.
3. Ground från arduino skall kopplas via en kondensator innan den kopplas i ground på wifi modulen enligt schemat nedan. Ground skall även kopplas till den sista resistorn.
4. 5V på arduinon kopplas med 2 dioder emellan innan den skall gå in i kondensatorn och kopplas till CH_PD och VCC porten på wifi modulen.
5. I/O port 3 skall kopplas till slutet på resistorerna som är seriekopplade. En resistor efter detta kopplas en kabel till RX på wifimodulen enligt schemat



Setup av Wi-Fi modulen

SoftwareSerial men detta bibliotek stödjer inte en BAUD hastighet högre än 9600. Därför behöver vi ändra BAUD hastigheten på ESP8266. Detta görs enligt följande steg nedan.

1. Sladden som sitter i pin 3 ska sitta i pin 0 (RX)
2. Sladden som sitter i pin 2 ska sitta i pin 1 (TX)
3. Ladda ett tomt program till Arduinon

4. Öppna Serial Monitor

5. Skicka in kommandot: "AT+GMR"
   Detta för att kontrollera vilken "AT firmware" som finns inprogrammerad på wi-fi modulen.
   Firmware är mjukvara som är inprogrammerad oftast på ROM eller flash minne.

6. Skicka in kommandot "AT+CIOBAUD=9600"

7. För att koppla upp sig mot sitt Wi-Fi börjar man med att ladda upp ett tomt program till arduinon från dess IDE.

8. För att ställa in rätt operationsläge börjar man med att skriva in AT. Detta kontrollerar att ESP8266 är okej. Denna ska ge "OK" om allting fungerar som det ska.

9. Skriv in "AT+RST". Detta startar om Wi-Fi modulen.

10. Skriv in "AT+CWMODE=1". Ställer in rätt operationsläge på modulen.

11. För att nu koppla upp sig mot Wi-Fi börjar vi med att skriva "AT+CWLAP" som listar alla tillgängliga nätverk.

12. Skriv sedan in "AT+CWJAP="SSID","PASS" där SSID är namnet på ert nätverk och PASS är lösenordet till nätverket.

13. Skriv till sist "AT+CWJAP?" för att visa vilket nätverk modulen är kopplad till, detta borde vara ditt nätverk.

14. Klart! Sätt tillbaka sladdarna som de satt innan

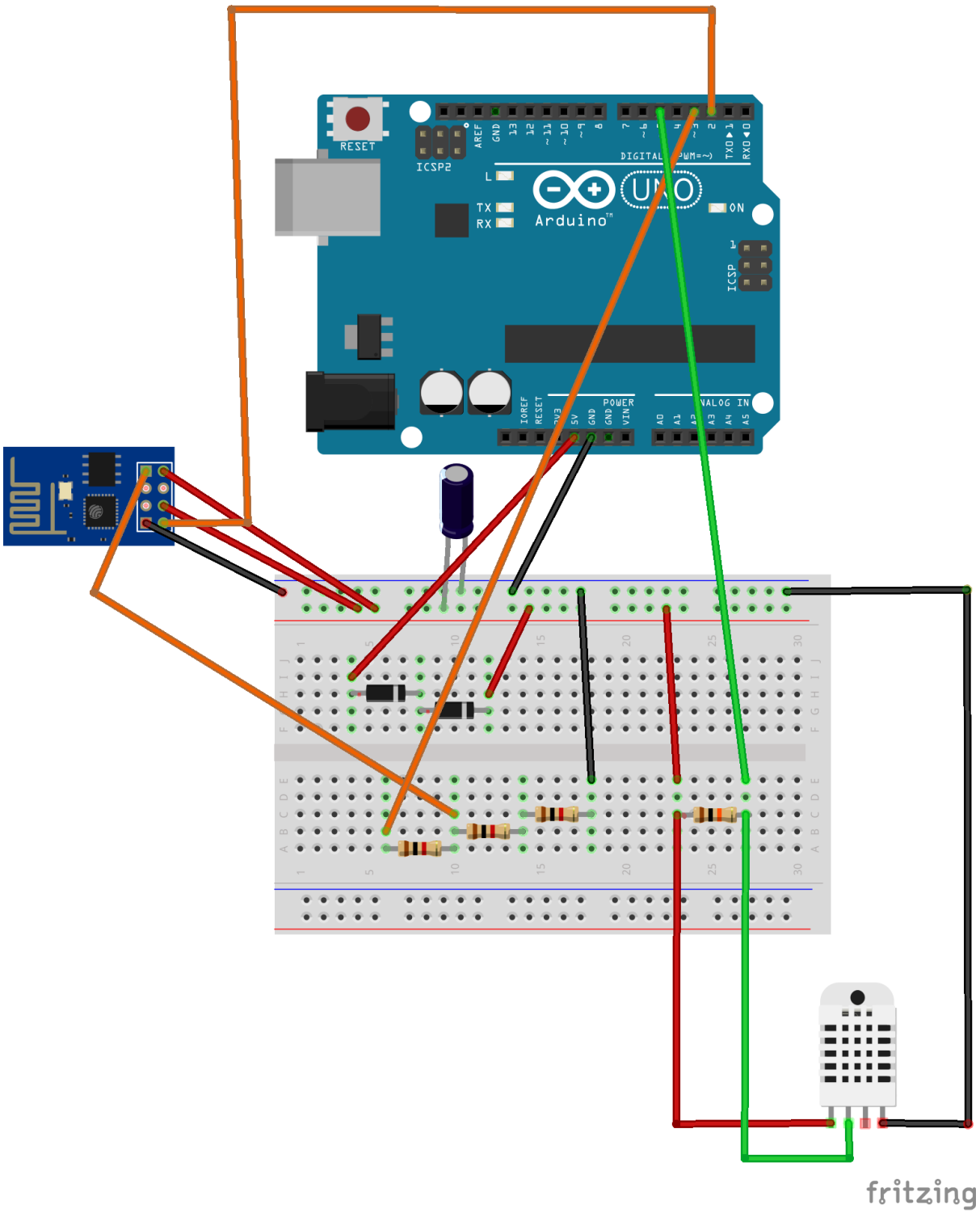
Tips! AT-kommandon är instruktioner som används för att kontrollera ett modem (i vårt fall

ESP8266?). AT är en förkortning för ATtention. Alla kommandon börjar med "AT".

Skicka temperatur och fuktighet till thingspeek med ESP8266 och DHT22
Nu ska vi beskriva hur ni kan skicka upp temperatur och fuktighetsmätningar till thingspeak.

Koppla kretsen

Både temperatur och fuktighetssensorn och wi-fi modulen ska vara in kopplade i Arduinon. Se till att strömadaptern används för att säkerställa att de båda modulerna får tillräckligt med ström.



Koda

Vi börjar med att definiera variabler och importera biblioteken som behövs. Vi gör även det som behövs i setup metoden.

```
#include <SoftwareSerial.h>
SoftwareSerial esp8266(2, 3); //RX, TX Defines what pin writes and reads data from arduino
#define IP "184.106.153.149" //thingspeak.com
```

```
String apiKey = "ERAN NYCKEL HÄR"; //get this from thingspeak

void setup(){
        esp8266.begin(9600); //esp baudrate is 9600
}
```

Använd koden för DHT sensorn, som vi gjorde innan, och koden för Wi-Fi modulen, koden ovan, i samma skript. Lägg även till de två nedanstående metoderna.

SendData metoden används för att skicka kommando till Wi-Fi modulen.

```
String sendData(String command, const int timeout)
{
        String response = "";
        esp8266.print(command); // send the read character to the esp8266
        long int time = millis();
        while ( (time + timeout) > millis())
        {
                while (esp8266.available())
                {
                        // The esp has data so display its output to the serial window
                        char c = esp8266.read(); // read the next character.
                        response += c;
                }
        }
        return response;
}
```

Metoden updateThingSpeak skickar data till thingspeek.

```
void updateThingSpeak(){
        sendData("AT+RST\r\n", 2000);//restarting wifimodule
        sendData("AT\r\n", 8000);
        // Setting up TCP connection
        String cmd = "AT+CIPSTART=\"TCP\",\"";
        cmd +=IP;
        cmd += "\",80";
        cmd += "\r\n";
        sendData(cmd, 4000);
        delay(5000);
        //Preparing to send data to thingspeak
        String getStr = "GET /update?api_key=";
        getStr += apiKey;
        getStr += "&field1=";
        getStr += String("ER DATA SOM SKALL SKICKAS UPP");
        getStr += "&field2=";
        getStr += String("NÄSTA DATA SOM SKALL SKICKAS UPP");
        getStr += "\r\n\r\n";
```

```
        //Sending the length of the data to be sent
        cmd = "AT+CIPSEND=";
        cmd += String(getStr.length());
        cmd += "\r\n\r\n";
        sendData(cmd, 4000);
        sendData(getStr, 4000);
        delay(6000);
}
```

Timers

För att vi ska kunna göra kontroller av växthuset med jämna mellanrum behöver vi använda en timer. I koden till växthuset använder vi oss av ett bibliotek som heter "Simple Timer". Detta kan hittas för nedladdning här: http://playground.arduino.cc/Code/SimpleTimer. Denna fil skall läggas i samma mapp som ditt skript.

```
#include "SimpleTimer.h"

// the timer objecta
SimpleTimer timer;
```

Med hjälp av timern kan vi starta metoder efter en viss tid eller köra en metod i ett intervall, detta kan göras med metoden timer.setInterval(). Vi skapar en metod som heter continuousCheck() som håller koll på temperatur och fuktihet och reglerar växthuset därefter. Denna metod vill vi köra var tionde minut så vi skapar ett intervall i setupmetoden med hjälp av setInterval().

```
void setup(){
        timer.setInterval(600000, continuousCheck); //600000 millisekunder = 10 minuter
}

void continuousCheck() {
        const int tempLimit = 22;
        const int highTempLimit = 24;

        const int humidMaxLimit = 65;
        const int humidMinLimit = 45;

        float tempDiff = fTemp - tempLimit;
        float highTempDiff = fTemp - highTempLimit;
        float humiditydMaxDiff = humidMaxLimit - fHumidity;
        float humidityMinDiff = fHumidity - humidMinLimit;
```

```
        Serial.print("Check Uptime (s): ");
        Serial.println(millis() / 1000);
        //fetch new data from sensors
        readSensor();

        if ((fTemp < tempLimit)) {
                Serial.println("Low temp");
                Serial.println("Temp Diff: " + String(tempDiff));
                closeWindow();
        }
        if ((fTemp > highTempLimit) || (fHumidity > humidMaxLimit)) {
                Serial.println("Temp or Humid high");
                Serial.println("Diff in temp: " + String(tempDiff));
                Serial.println("Diff in humidity: " + String(humiditydMaxDiff));
                openWindow();
                startFan();
        }
        else if ((fTemp > tempLimit)) {
        Serial.println("High temp");
        Serial.println("Diff in temp: " + String(tempDiff));
        openWindow();
        startFan();
        }
        if (fHumidity < humidMinLimit) {
        Serial.println("Low humid");
        Serial.println("Diff in humidity: " + String(humidityMinDiff));
        startPump();
        }
        waterCheck();
        Serial.println("---");
        count++;
}
```

I loopen så startar vi timern genom följande kod:

```
void loop(){
        timer.run(); //start the timer
}
```

Process logik

För att minimera risken att överbelasta arduinon använder vi oss av två metoder för att göra så att det endast är en process som körs åt gången. Dessa är pausProcess() och continueProcess(). Om Wifi modulen inte används behövs endast pausprocess köras innan pumpen startas för att förhindra att pumpen och fläkten körs samtidigt. Om Wifi modulen används behöver pausprocess även köras när Wifi modulen körs.

```
/*
   1 = pump, 2 = ESP8266
*/
void pausProcess(int typeOfDevice) {
        if (typeOfDevice == 1) {
                if (fanStatus) {
                        stopFan();
                        continueFan = true;
                }
        }
        if (typeOfDevice == 2) {
                if (fanStatus) {
                        stopFan();
                        continueFan = true;
                }
                if (pumpStatus) {
                        stopPump();
                        continuePump = true;
                }
        }
}

void continueProcess() {
        if (continueFan) {
                startFan();
                continueFan = false;
        }
        if (continuePump) {
                startPump();
                continuePump = false;
        }
}
```

Dessa två metoder behöver nu köras i updateThingSpeak metoden för att pausa processerna och sedan starta dem igen. Längst upp skriver vi:

```
void updateThingSpeak(){
        pausProcess(2);
```

och i slutet skriver vi

```
        continueProcess()
```

Vi behöver även lägga till dessa i startPump och stopPump metoderna.

```
void startPump() {
        pausProcess(1);
        digitalWrite(pumpPin, HIGH);
}
void stopPump() {
        digitalWrite(pumpPin, LOW);
        lastWateredMillis = millis();
        continueProcess();
}
```

För att inte övervattna växterna skapar vi en metod som håller koll på när dem senast vattnades.

```
void waterCheck() {
        unsigned long currentMillis = millis();
        if(currentMillis - lastWateredMillis >= wateringInterval) {
                startPump();
        }
}
```

Hela koden

För att se hela koden sammansatt i en fil kan ni gå till hemsidan och ladda ner den fullständiga koden.

Observera

Ni kan nu observera växthuset och se hur det fungerar. Prova även att plantera lite olika frön och se om de börjar växa. Bra växter att plantera kan vara olika kryddor då dessa växer relativt snabbt och man kan se resultat inom några dagar. Viktigt att tänka på när ni planterar är att blöta jorden innan så den blir lätt fuktig. Detta gör att det räcker med mindre bevattning senare.

Efter några dagar kan ni ställa er frågan om växthuset fungerar som det ska eller om ni behöver justera några värden. Har saker börjat växa eller syns det inget? Om det inte växer något kan det vara bra att kolla upp vad växterna ni planterat behöver ha för miljö eller om dessa ens lämpar sig för ett växthus. Det kan vara så enkelt som att temperaturen behöver vara lägre eller att växten är känslig för övervattning.

Utökning av växthuset

Börja med att lägga till Wi-Fi modulen som är valfri och prova att skicka upp data till thingspeak. Om ni vill göra en egen hemsida som ni designar själva och visar information om växthuset på kan ni koppla er arduino som sköter växthuset till en Raspberry Pi t.ex. Med hjälp av den kan ni läsa data från arduinon och skapa en webserver med hjälp av t.ex. Python. Sedan kan ni skapa eran hemsida med hjälp av HTML, CSS och Javascript.

Ni skulle även kunna fundera på utökningar av själva växthuset, fler sensorer t.ex. Hade det varit intressant att mäta fuktigheten i jorden? Vad hade det gett för fördelar? Nackdelar? Ni kan även ställa er frågan om en lampa för att simulera solen kan vara nödvändigt om det är mörkare. Vad finns det för för och nackedelar med en lampa?

Ni kan också fråga er om det finns något bättre sätt att utforma växthuset på. Går det att vattna på ytterligare sätt? Kan man fånga upp spillvatten som missar krukorna vid takbevattning? Kan man mäta flera saker och justera andra saker för att ge växterna en ännu bättre miljö? Kanske en arm som gödslar en gång i veckan.

Var inte rädda för att prova andra saker med växthuset och förbättra det, det är lite poängen med det att ni ska komma på förbättringar. Tänk på att det finns inga gränser.

Frågor att ställa er själva innan ni börjar

- **Hur fungerar ett växthus?**
- **Vilka sensorer behövs?**
- **Hur ska bevattningen ske? Tak? Vid mark?**
- **Räcker strömstyrkan från arduino?**
- **Har vi kopplat rätt?**
- **Vilka växter ska vi plantera? Vilken miljö trivs dem i?**