[**Minecraft Installation guide (mac)**](#_a5aodah5eg3) **2**

[Setting up a modded Minecraft server](#_vlphdo6jguqz) **2**

[TnT Craft](#_ksapr8idwsux) **3**

[Arduino connection](#_4p2p1agefmq8) **3**

[Electrical Age](#_k1008wviv48h) **3**

[Basics](#_wqzvgbel8dix) 3

[Electronics](#_tlhbqh5fqaum) 4

[Electrical Probes and monitors](#_xy6911uj31y6) 4

[Serial Craft connection to Wind turbine (WT)/Hydro turbine (HT)](#_7h1cdt9c54hf) 5

[**Appendix**](#_gunptp9dbebv) **6**

[Arduino](#_uxaiqonz17j9) 6

[Arduino Code](#_4jtho1acds5n) 7

**This** [**YouTube video**](https://www.youtube.com/watch?v=0afYwq3p4G0&feature=youtu.be) **can be used as a guide.**

# 

# Minecraft Installation guide (mac)

1. Download and install Minecraft: (Ask your teacher (Dan) to buy a license) <https://my.minecraft.net/nl-nl/store/minecraft/#register>
2. Download all the files from [“Download Files”](https://drive.google.com/open?id=1Vlb9kM5kNzh6NL6VZQLDUWg3kFnIzCxL)
3. Run: forge-1.7.10-10.13.4.1566-1.7.10-installer.jar and select “install client”.

Close windows when finished.

1. Open Minecraft Launcher
   1. In the bottom left, select “Forge 1.7”. If this is not an option, go to: “installations” at the top of the window, add in “version” select “Forge 1.7.10 (…)” and give any name.
   2. Start the game by clicking on “Play”.
2. Once Minecraft has launched click Esc and quit the game.
3. Go to “Finder” and look for this folder: **/Users/[user]/Library/Application Support/minecraft/mods** (Use cmd+shift+G when in Finder to directly go to this folder).

(On windows, go to ‘this computer’ and search for “.minecraft”)

1. Move the downloaded mod files (Electrical age, Serialcraft, TickrateChanger and jssc, from **step 2**) into this folder. “jopt-simple-4.5” will be used in step 8.
2. **Replace** the “jopt-simple-4.5.jar” file in

**/Users/[user]/Library/Application Support/minecraft/libraries/net/sf/jopt-simple/jopt-simple/4.5**

with the “jopt-simple-4.5” file from the [“Download Files”](https://drive.google.com/open?id=1Vlb9kM5kNzh6NL6VZQLDUWg3kFnIzCxL)

1. Open Minecraft and start up the game (step 4)
2. Click on Mods to check if all mods are installed properly:
   1. Forge mod loader
   2. Minecraft Forge
   3. Electrical Age
   4. Serialcraft
   5. Tickrate Changer
3. Start a single player world or join a multiplayer world

# Setting up a modded Minecraft server

* Make sure you also add the mods in the ‘mods ‘ folder in server 1.7
* Don’t forget to start the server.bat file (see video: .bat file needed for modded server)
* **Go to the config folder and open Eln. Scroll down to Gameplay and change Explosions to False to prevent exploding electrical systems.**

<https://www.youtube.com/watch?v=7J9aq4fV_GE>

## 

# TnT Craft

The following steps assume that you have already made a single player world in Minecraft.

1. Close Minecraft (and your server)
2. Download the [world folder](https://drive.google.com/open?id=1qRz3cP8uS06ggWzS97qPOqLjDUfoICZv)
3. Add this folder to: **/Users/[user]/Library/Application Support/minecraft/saves**

(Use cmd+shift+G when in Finder to directly go to this folder).

1. You can now open this folder in your Minecraft Singleplayer menu. For server world, continue to step 5.
2. Repeat steps 1-3 for your Minecraft server folder. Instead of the “**saves**” folder, replace the “**world**” folder with the downloaded “**world**” folder. Make sure the new folder is (re)named to “**world**”
3. Start up Minecraft and server.

# Arduino connection

This description assumes that you have used Arduino before.

1. Download the Arduino [Serial Craft folder](https://drive.google.com/open?id=1S-wDuJ76240E0Cx-0mpnrf1UaBCZVIbG)
2. Include the library into Arduino by using these [instructions](https://www.arduino.cc/en/Guide/Libraries#toc4)
3. Download the [Arduino code](https://drive.google.com/drive/folders/1_2g5i4ZSSlsnh7McRUbcuee5adEQdAuY?usp=sharing) and connect Arduino using the diagram below.
4. When in the Minecraft world, press K to open up the connection screen and select the appropriate port for your Arduino. The command from the Arduino will now appear in the text field in Minecraft. The DC motor will turn on and rotate at a speed based on the amount of wind energy produced in Minecraft.

* add comments in arduino

# Electrical Age

## Basics

* The goal will be to supply all the lights with power. The whole system works at **200V**.
* All solar panels output 12,5 volt (press shift while hovering your mouse over an item to see details).

You will need to make arrays of 4 panels (50V) and convert 50V to 200V with a dc/dc-converter (or place 16 panels in series). You will have to insert 4 medium voltage cables, one low voltage cable and a ferromagnetic core into the converter (do it in this order to prevent explosions). Now you can use medium voltage cables to connect the converted power to utility poles and bridge long distances without big losses of power. The utility poles (with converter) will automatically quadruple the voltage and divide by 4x at the beginning and end of the HV network respectively.

* The same can be done for wind turbines, but you can only connect in the wind turbines in parallel not 4 in series since wind turbines already give 50V. Same applies to water turbines.
* All the energy producers are connected to the grid with the utility poles (see example in world). The grid starts and ends with Utility poles with converter. The connection extenders are normal utility poles.   
   In the storage house, the poles are connected to arrays of 4x50V = 200V *Current Oriented Batteries* that will be charged during the day and discharged during the night. The power input goes into the RED side of the battery and a ground must be connected to the BLACK side in order for them to work.
* **IMPORTANT**: When choosing cables, pay attention to the max voltage, power and current it can withstand. Choose your grid setup wisely. We used 200V because the DC/DC-converters increase voltage which results in a lower amperage.

## Electronics

* The power output from all the batteries is connected to the so called ‘control room’.

There, all the different lights are managed with **lamp** **supplies** that are connected to light sensors or switches. Lamp supplies allow for lamps to be powered without needing a physical cable connection (not realistic, but convenient). Lamp supplies need to be configured to a channel. Lamp sockets that are placed in the MC world must be configured to the same corresponding channel. When placing lamps, make sure you configure your first lamp to the preferred channel, the following lamps will be configured to the same channel. Don’t forget to supply the lamps with (the correct) light bulbs and cables. Supplying the Lamp Supply with more cables increases the range.

Connect a ‘medium voltage switch’ to the lamp supply to manually switch the light on/off.

If you do want them to change with the time of day, you must connect a relay between the input and the lamp supply and connect a daylight sensor to the relay with a signal cable.

Right click on the relay and make sure it’s on “normally closed”. If the Relay and daylight sensor don’t connect to the signal cable, press **C + Rightclick** to turn the devices (this can be used on most blocks).

#### Electrical Probes and monitors

In order to connect a monitor to a cable, you must first add an electrical probe into the cable (and add a cable into the electrical probe). Right click on the probe and configure the values that you need in order to output a 100% and 0% value. Don’t forget to select the correct unit on the left.

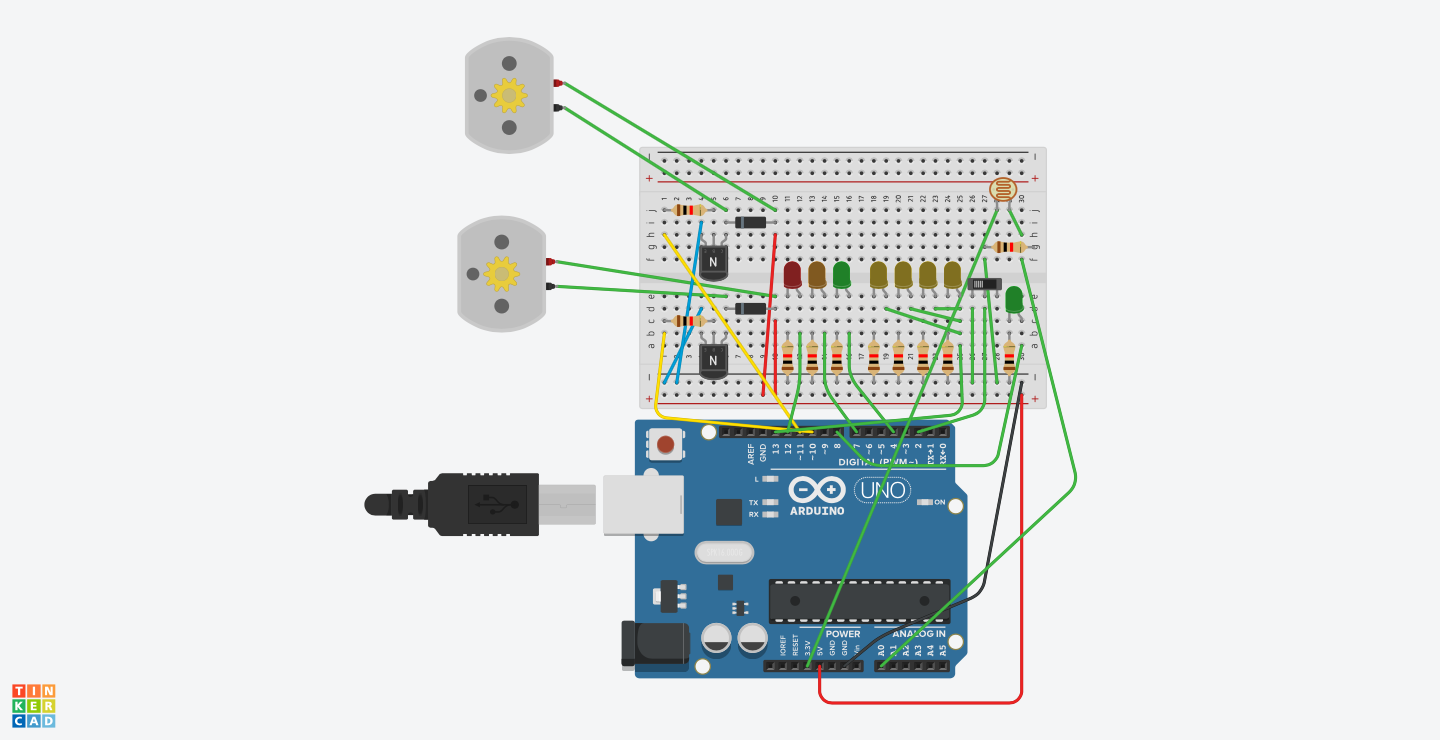
Connect the probe to the data logger. To display the correct y-axis, right click on the data logger and click on configure. Select the preferred unit and values and press Enter on your keyboard.

#### Serial Craft connection to Wind turbine (WT)/Hydro turbine (HT)

In our demo, the speed of the Arduino WT and HT were based on the power production of each of the power sources. To do this, connect an electrical probe to the cable of WT or HT. Next, right click on the probe and select power. Enter the 100% and 0% values, based on the amount of turbines placed and the information from the inventory (holding shift). Connect this with a voltage to redstone converter, add **one** redstone and place a “Redstone Serial Message” Block. When placing this block it will prompt a box to type an ID which will correspond with the ID in Arduino. The probe will convert the output value into a redstone value between 0-15. This value will be mapped in Arduino from 0-255 to control the DC motors.

# Appendix

## Arduino



## Arduino Code

#include <SerialCraft.h>

SerialCraft sc;

const int ledPinHouse1 = 13; //Lantern houses

const int ledPinSolar = 8; //Solar production

const int ledPinGreen = 4; //Storage green

const int ledPinOrange = 7; //Storage orange

const int ledPinRed = 12; //Storage red

const int redstoneDC1Pin = 10; //Motor pin WT

const int redstoneDC2Pin = 11; //Motor pin HT

void setup() {

sc.setup();

// Arduino LED control

sc.registerDigitalInputCallback(sendMessage, 2); // Serialcraft command to send message based on signal from digital port 2

sc.registerAnalogInputCallback(analogCallback, A0); // Serialcraft command to send message based on signal from analog port A0

sc.registerRedstoneSerialCallback(redstoneSerial); // Serialcraft command to start RedstoneSerialMessage block connection

pinMode(ledPinHouse1, OUTPUT); // Set pin to output

pinMode(ledPinSolar, OUTPUT); // Set pin to output

pinMode(ledPinRed, OUTPUT); // Set pin to output

pinMode(ledPinOrange, OUTPUT); // Set pin to output

pinMode(ledPinGreen, OUTPUT); // Set pin to output

// Serial DC Control

pinMode(redstoneDC1Pin, OUTPUT);

pinMode(redstoneDC2Pin, OUTPUT);

}

void loop() {

sc.loop();

}

// Serial DC Control to map the input from the turbines in Minecraft,

// that will make the dc motors rotate in variable speeds

// Each READING will refer to their own command from Void Setup

void redstoneSerial(int s, String id) {

if(id == "WT") { // Will retrieve value from RedstoneSerialMessage block with ID WT

analogWrite(redstoneDC1Pin, map(s, 0, 15, 0, 220)); // Maps 0-15 value to 0-255

} if(id == "HT") { // Will retrieve value from RedstoneSerialMessage block with ID HT

analogWrite(redstoneDC2Pin, map(s, 0, 15, 0, 200)); // Maps 0-15 value to 0-255

}

}

// The on/off switch will send a signal to Minecraft that will create a chat message.

// The chatmessage will send the command for clear weather or rainy weather depending on the state of the switch.

void sendMessage(bool reading) {

if(reading == HIGH) {

sc.sendChatMessage("/weather clear");

}

if(reading == LOW) {

sc.sendChatMessage("/weather rain");

}

}

// Light sensor

// The light sensor will read a value between 50 and 625 depending on real world brightness (Adjust to your own setting with the assist of Serial Monitor)

// Using this you can make it day and night in Minecraft through the chat message

// LEDs in the real world are also connected and will go on and off simultaneously with the lights in minecraft

void analogCallback(int reading) {

int t = map(reading, 50, 625, 18000, 6500);

sc.setTime(t);

// This will control the lights on the Arduino (works around Minecraft, can be done with RedstoneSerialMessage)

if (reading < 450) {

digitalWrite(ledPinHouse1, HIGH);

}

if (reading > 450) {

digitalWrite(ledPinHouse1, LOW);

digitalWrite(ledPinGreen, HIGH);

digitalWrite(ledPinOrange, LOW);

digitalWrite(ledPinRed, LOW);

}

if ((reading > 300) && (reading < 450)) {

digitalWrite(ledPinGreen, LOW);

digitalWrite(ledPinOrange, HIGH);

digitalWrite(ledPinRed, LOW);

}

if (reading < 300) {

digitalWrite(ledPinSolar, LOW);

digitalWrite(ledPinGreen, LOW);

digitalWrite(ledPinOrange, LOW);

digitalWrite(ledPinRed, HIGH);

}

//This will turn on the Solar cell LED (same workaround as stated above)

if (reading > 343){

digitalWrite(ledPinSolar, HIGH);

}

}