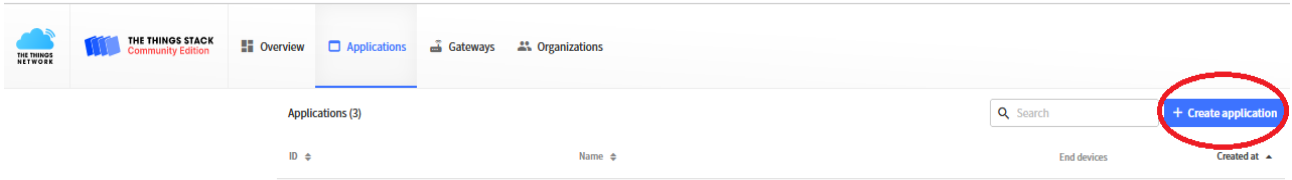
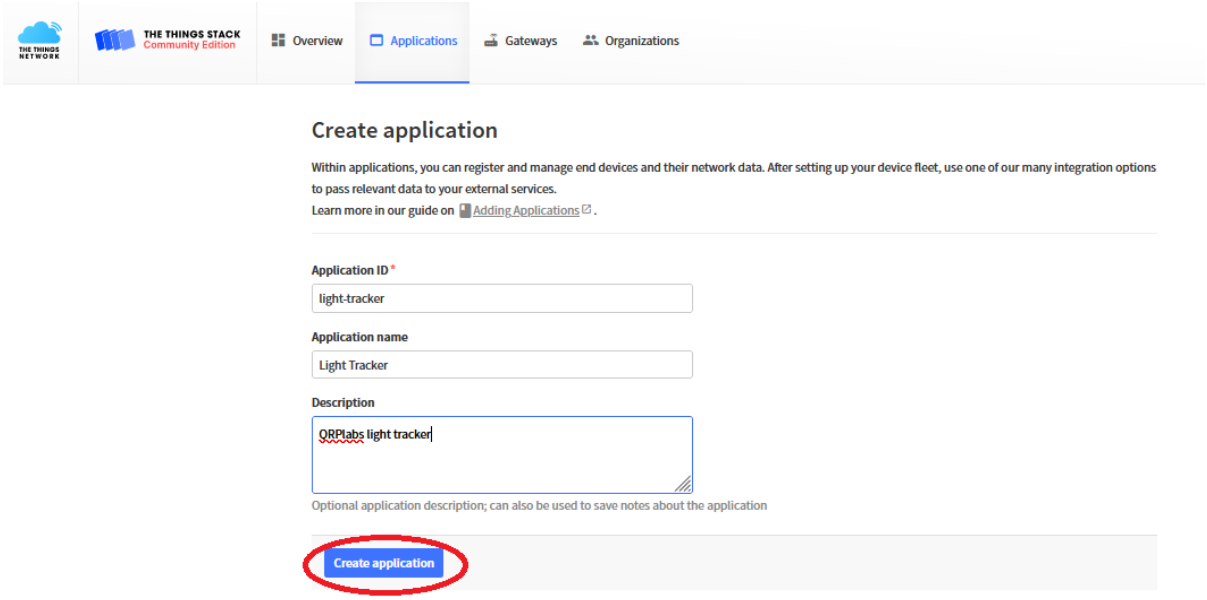
Onbarding a Light Tracker with TTN

Create TTN account

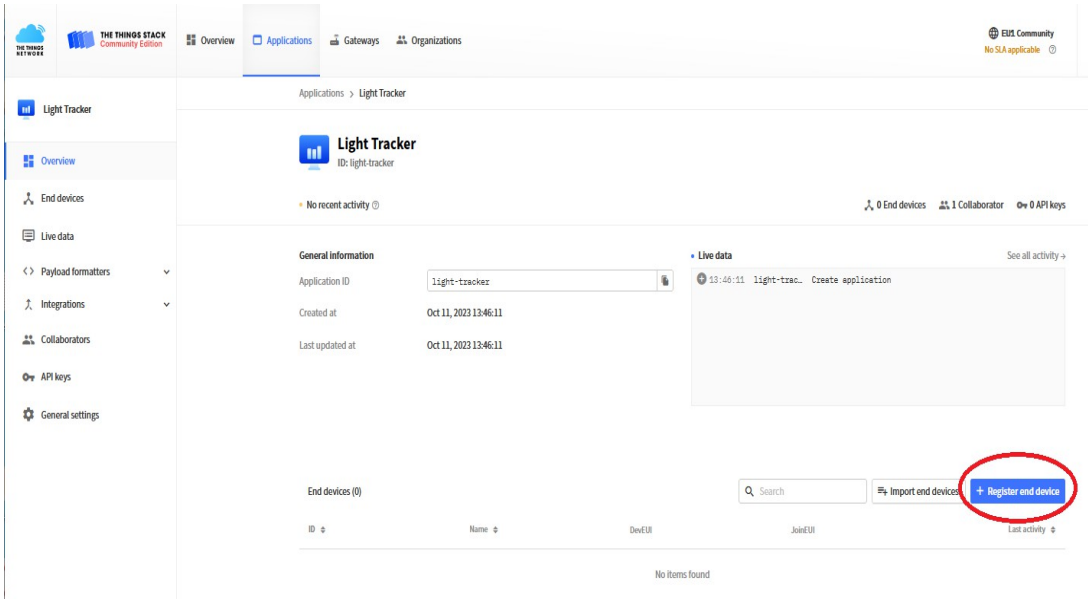
Create a Application



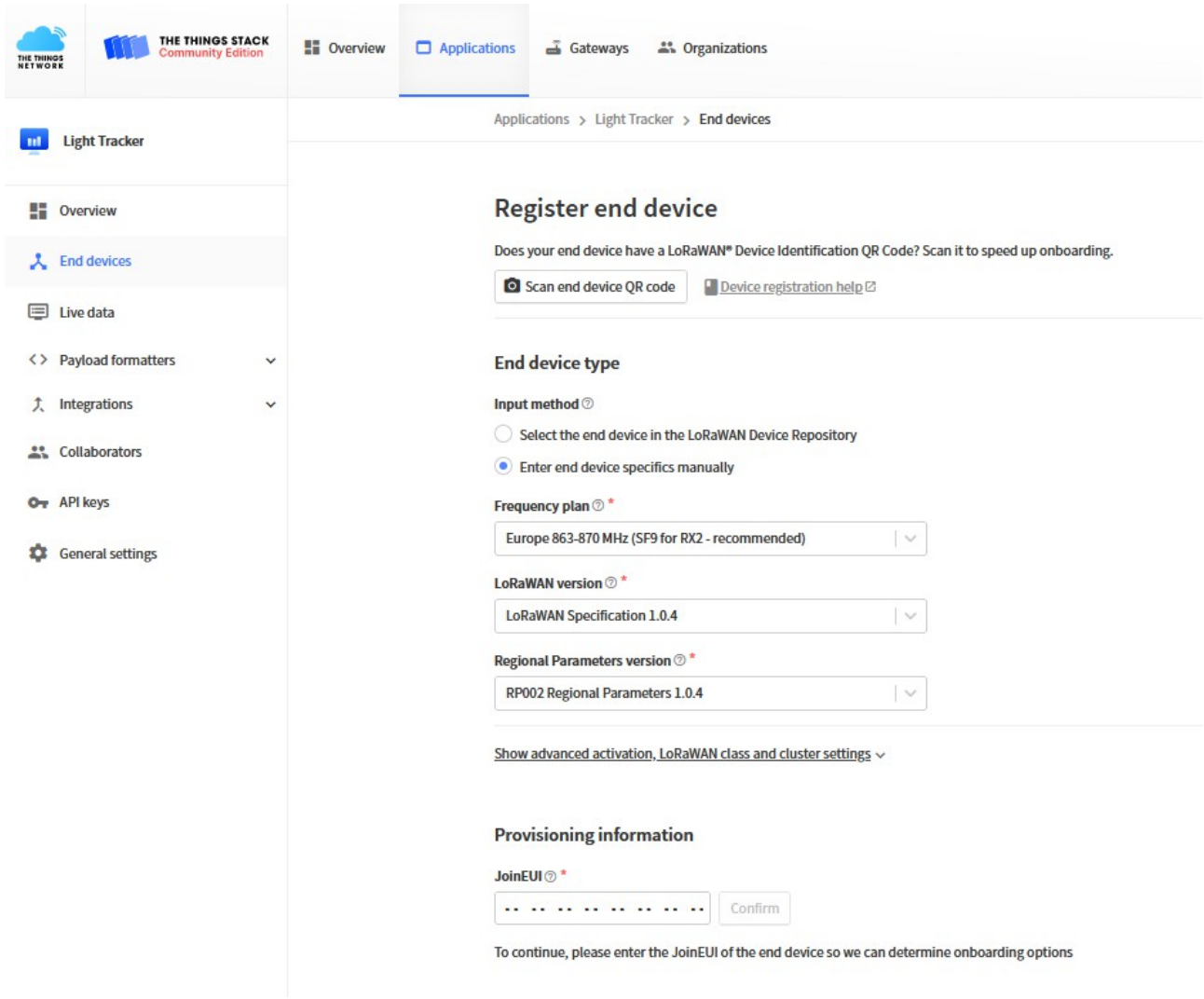
and give it a name



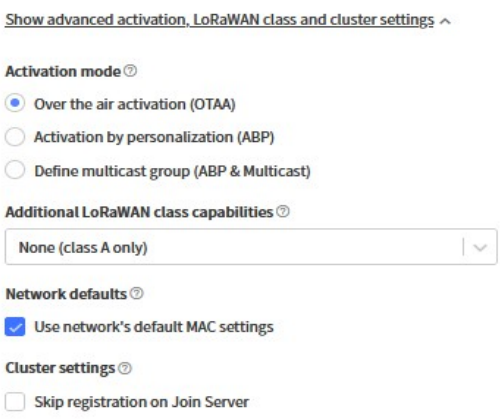
And register a new device:



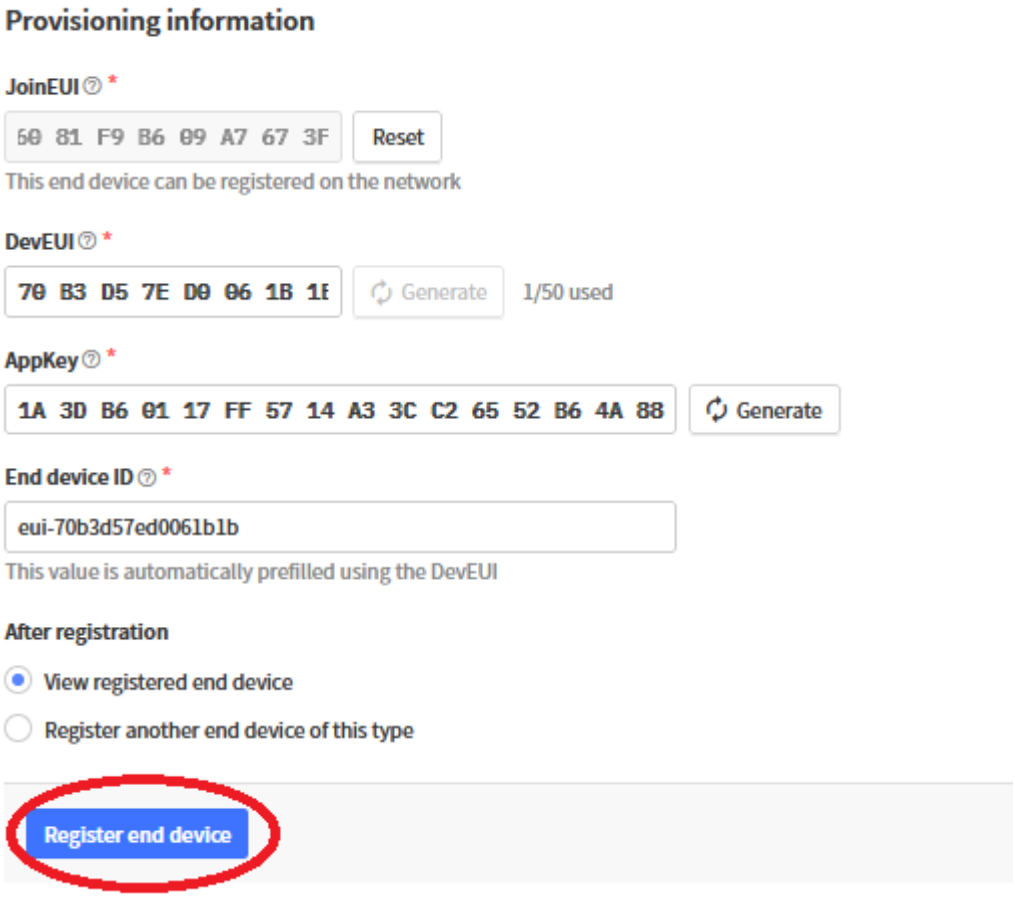
the Light tracker is a LoraWAN 1.0.4 device and you will need to set up a initial Frequency Plan.



Click on Show advanced activation, LoRaWAN class and cluster settings and select Over the Air Activation and Class A LoraWAN



JoinEUI for the light tracker is 6081F9B609A7673F and generate the Device EUI and Appkey:



and register the device.

Use the generated keys in the Light Tracker software, compile and program:

```
//***************************** UPDATE HERE WITH YOUR DEVICE KEYS ***********************************/
//You should copy device keys from Helium or TTN Console and update following keys. Please check out:
https://github.com/lightaprs/LightTracker-1.0/wiki/Adding-Device-on-Helium-Console

// This EUI must be in little-endian format, so least-significant-byte (lsb)
// first. When copying an EUI from Helium Console or ttnctl output, this means to reverse the bytes.
static const u1_t PROGMEM DEVEUI[8]={0x1B, 0x1B, 0x06, 0xD0, 0x7E, 0xD5, 0xB3, 0x70}; //ttn

// This EUI must be in little-endian format, so least-significant-byte (lsb)
// first. When copying an EUI from Helium Console or ttnctl output, this means to reverse the bytes.
static const u1_t PROGMEM APPEUI[8]={0x3F, 0x67, 0xA7, 0x09, 0xB6, 0xF9, 0x81, 0x60}; //ttn

// This key should be in big endian format (or, since it is not really a
// number but a block of memory, endianness does not really apply). In practice, a key taken from Helium Console or ttnctl can be
copied as-is.

static const u1_t PROGMEM APPKEY[16] = {0x1A, 0x3D, 0xB6, 0x01, 0x17, 0xFF, 0x57, 0x14, 0xA3, 0x3C, 0xC2, 0x65, 0x52, 0xB6, 0xA4,
0x88}; //ttn
```