This weeks challenge is to build a working clock.

This is not just an ordinary clock though, we are going to build a clock that works in the same way as the Railway clocks works in Switzerland.

These clocks work in the following way:

- The Clock waits for a signal from the master clock
- When it gets the signal, it moves the minute hand on by one minute (and the hour hand if necessary)
- then the second hand moves round once in about 58 seconds
- It then waits for the signal again from the master clock.

This means that all the clocks on all the Stations are synchronised exactly.

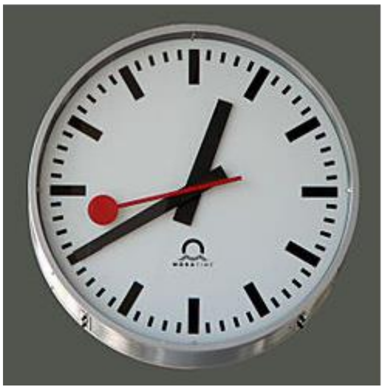
In Computer speak this is called Event Driven Programming, where the code simply waits for an Event before doing anything.

So what's the plan?

We need 4 Sprites

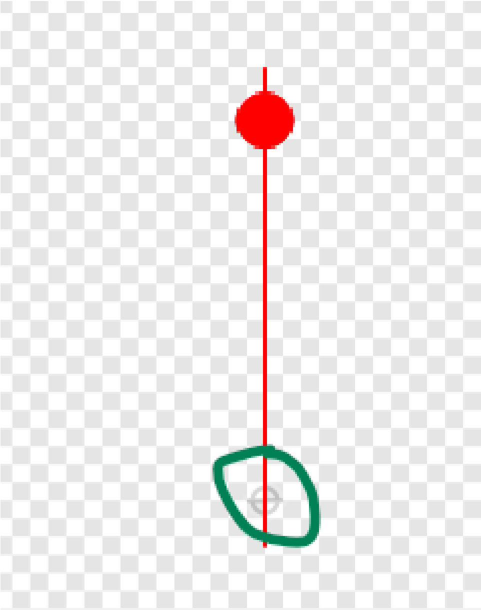
1. For the Clock Face
2. For the Hour Hand
3. For the Minute Hand
4. For the Second Hand

You can create them however you want but this is what the Swiss railway clocks look like and I will try and create something similar.



Remember one thing, when creating the Sprites make sure they are centred correctly.

So, the clock face should be perfectly centred and then the hands should be centred near one end something like this, I have highlighted where the centre is, with the green circle.



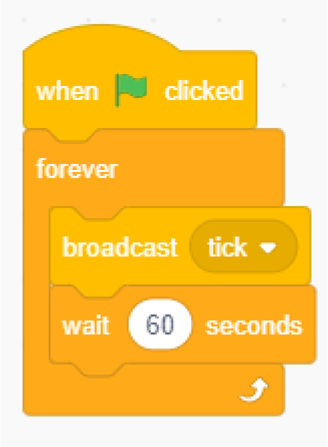
The code

Remember this will be event driven, so we will be using Broadcasts a lot. We will also have to calculate how far to move each hand, this means we will be using the number 360 a lot as that is one complete circle.

So just to let you know how far each hand should move:

- The Second hand will move 360 degrees for each event.
- The Minute hand will move 6 degrees for each event (360/60 minutes)
- The Hour hand will move 0.5 degree for each event (360/60 minutes/ 12 hours)

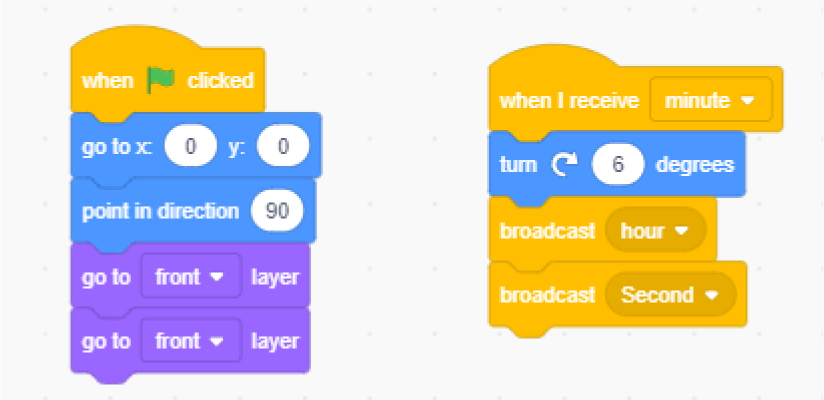
The first event will be broadcast from the Stage:



```
when ⚑ clicked
forever
    broadcast tick ▾
    wait 60 seconds
```

The Minute hand will receive this broadcast and move 6 degrees it will then broadcast another 2 events, one to the Hour hand and one to the second hand so they can move.

This is the code for the Minute hand which should be enough to work out the rest of the code as well.



```
when 🏴 clicked
go to x: 0 y: 0
point in direction 90
go to front ▾ layer
go to front ▾ layer
```

```
when I receive minute ▾
turn ↻ 6 degrees
broadcast hour ▾
broadcast Second ▾
```

The extra code above is to make sure the Minute hand starts in the correct position.

And one final thing which might be a little tricky, but I'll leave it as a challenge for you, the Second hand should only take 58 seconds to go all the way around.