```
--[[TreeFarm Stuff, for trees!]]--
function populateSeedTypeLookUpTable()
if seedTypeLookUpTable==nil then seedTypeLookUpTable = {} end
  for seedTypeName, seedType in pairs(global.tf.seedPrototypes) do
    for _, stateName in pairs(seedType.states) do
      seedTypeLookUpTable[stateName] = seedTypeName
    end
  end
end

script.on_init(function()
	if not remote.interfaces["treefarm_interface"] then
		debug("Treefarm not installed")
		Trees.OnInit()
	elseif remote.interfaces.treefarm_interface and
remote.interfaces.treefarm_interface.getSeedTypesData then
		remote.call("treefarm_interface", "addSeed", Trees.RubberTree)
		remote.call("treefarm_interface", "addSeed", Trees.SulfurTree)
	end
	fs.Startup()
	fs.World_Call()
	fs.Wind_Startup()
end)

script.on_load(function()
	if not remote.interfaces["treefarm_interface"] then
		Trees.OnLoad()
	elseif remote.interfaces["treefarm_interface"] and
remote.interfaces.treefarm_interface.getSeedTypesData then
		if not remote.call("treefarm_interface", "readSeed", "RubberTree") then
			remote.call("treefarm_interface", "addSeed", Trees.RubberTree)
		end
		if not remote.call("treefarm_interface", "readSeed", "SulfurTree") then
			remote.call("treefarm_interface", "addSeed", Trees.SulfurTree)
		end
	end
	if not global.Wind then fs.Wind_Startup() end
end)

script.on_event(defines.events.on_tick, function(event)

	fs.Timer(event)
	if event.tick%600==1 then
		GUI.CreateButton()
	end
	if not remote.interfaces["treefarm_interface"] then
```

```
	while ((global.tf.growing[1] ~= nil) and (event.tick >= global.tf.growing[1].nextUpdate))
do
  local removedEntity = table.remove(global.tf.growing, 1)
  local seedTypeName
  local newState
	if seedTypeLookUpTable==nil then populateSeedTypeLookUpTable() end
		if removedEntity.entity.valid then
			seedTypeName = seedTypeLookUpTable[removedEntity.entity.name]
			newState = removedEntity.state + 1
			if newState <= #global.tf.seedPrototypes[seedTypeName].states then
			local tmpPos = removedEntity.entity.position
			local newEnt = game.get_surface("nauvis").create_entity{name =
global.tf.seedPrototypes[seedTypeLookUpTable[removedEntity.entity.name]].states[newStat
e], position = tmpPos}
			removedEntity.entity.destroy()
			debug("Old tree removed, new one placed")
			local deltaTime = math.ceil((math.random() *
global.tf.seedPrototypes[seedTypeName].randomGrowingTime +
global.tf.seedPrototypes[seedTypeName].basicGrowingTime) / removedEntity.efficiency)
			local updatedEntry =
			{
				entity = newEnt,
				state = newState,
				efficiency = removedEntity.efficiency,
				nextUpdate = event.tick + deltaTime
			}
			Trees.placeSeedIntoList(updatedEntry)
			debug("seed placed into list (ontick event")
			end
		end
	end end
end)
script.on_event(defines.events.on_built_entity, function(event)
	fs.BuildEntityLogger(event.created_entity.name)
	fs.LoggerCount("PlayerBuilt", 1)
local player = game.players[event.player_index]
	if not remote.interfaces["treefarm_interface"] then
	if event.created_entity.type == "tree" then
	if seedTypeLookUpTable==nil then populateSeedTypeLookUpTable() end
	debug("tree created (player "..tostring(event.player_index)..")")
  local currentSeedTypeName = seedTypeLookUpTable[event.created_entity.name]
		if currentSeedTypeName ~= nil then
		debug("currentSeedTypeName ~= nil")
		local newEfficiency = Trees.calcEfficiency(event.created_entity, false)
```

```
                local deltaTime = math.ceil((math.random() *
global.tf.seedPrototypes[currentSeedTypeName].randomGrowingTime +
global.tf.seedPrototypes[currentSeedTypeName].basicGrowingTime) / newEfficiency)
                local nextUpdateIn = event.tick + deltaTime
                local entInfo =
                {
                        entity = event.created_entity,
                        state = 1,
                        efficiency = newEfficiency,
                        nextUpdate = nextUpdateIn
                }
                Trees.placeSeedIntoList(entInfo)
                debug("seed placed into list (player "..tostring(event.player_index)..")")
                return
                debug("return")
                end
        end end
end)

script.on_event(defines.events.on_robot_built_entity, function(event)
        fs.RobotBuildEntityLogger(event.created_entity.name)
        fs.LoggerCount("RobotBuilt", 1)
local player = game.players[event.player_index]
        if not remote.interfaces["treefarm_interface"] then
        if event.created_entity.type == "tree" then
        if seedTypeLookUpTable==nil then populateSeedTypeLookUpTable() end
        debug("tree created (Robot)")
   local currentSeedTypeName = seedTypeLookUpTable[event.created_entity.name]
                if currentSeedTypeName ~= nil then
                debug("currentSeedTypeName ~= nil")
                local newEfficiency = Trees.calcEfficiency(event.created_entity, false)
                local deltaTime = math.ceil((math.random() *
global.tf.seedPrototypes[currentSeedTypeName].randomGrowingTime +
global.tf.seedPrototypes[currentSeedTypeName].basicGrowingTime) / newEfficiency)
                local nextUpdateIn = event.tick + deltaTime
                local entInfo =
                {
                        entity = event.created_entity,
                        state = 1,
                        efficiency = newEfficiency,
                        nextUpdate = nextUpdateIn
                }
                Trees.placeSeedIntoList(entInfo)
                debug("seed placed into list (Robot)")
                return
                debug("return")
```

```
                end
        end end
end)
```