```
begin
    using CSV
    using DiffEqFlux
    using Flux: train!, Chain, Dense
    using Flux
    using DataInterpolations
    using DataFrames
    using OrdinaryDiffEq
    using Plots
end
```

```
celsius2kelvin (generic function with 1 method)
```

```
celsius2kelvin(c) = c + 273.15
```

" Training inzwischen viel schneller Änderung. Workbook auf geteilt eines mit nODE und eines mit PINN. zusätzlich Bibleotheken nicht in lokal gespeichten, sondern mit Pluto gemanagt.

Aktuelle Versionen der Bibleothek als zuvor!?

```
md"""
Training inzwischen viel schneller Änderung.
Workbook auf geteilt eines mit nODE und eines mit PINN.
zusätzlich Bibleotheken nicht in lokal gespeichten, sondern mit Pluto gemanagt.

Aktuelle Versionen der Bibleothek als zuvor!?
"""
```

```
prepare_dataframe (generic function with 1 method)
```

```
function prepare_dataframe(data_frame_path)

    _df = DataFrame(CSV.File(data_frame_path))

    start_times_stat = unique(_df, :run)

    _df.start_time =  start_times_stat[trunc.(Int, _df.run).+1, :time]
    _df.duration = _df.time - _df.start_time
    _df.T1 = _df.T1 .|> celsius2kelvin
    _df.T1prev = prepend!(_df.T1[1:end-1], _df.T1[1], )
    _df.Column1 = _df.Column1 .+ 1
    return _df
end
```

	Column1	T1	T2	Q1	time	run	start_time	duration	T
1	1	294.693	22.349	100.0	1.68496e9	0.0	1.68496e9	0.0	29
2	2	294.693	22.188	100.0	1.68496e9	0.0	1.68496e9	1.35462	29
3	3	294.693	22.316	100.0	1.68496e9	0.0	1.68496e9	2.5741	29
4	4	294.693	22.316	100.0	1.68496e9	0.0	1.68496e9	3.794	29
5	5	294.693	22.316	100.0	1.68496e9	0.0	1.68496e9	5.01308	29
6	6	294.693	22.413	100.0	1.68496e9	0.0	1.68496e9	6.23228	29
7	7	295.015	22.413	100.0	1.68496e9	0.0	1.68496e9	7.45149	29
8	8	295.015	22.381	100.0	1.68496e9	0.0	1.68496e9	8.67103	29
9	9	295.338	22.413	100.0	1.68496e9	0.0	1.68496e9	9.8905	29
10	10	295.338	22.381	100.0	1.68496e9	0.0	1.68496e9	11.1097	29
more									
8640	8640	298.238	26.377	0.0	1.68497e9	11.0	1.68496e9	868.37	29

```
begin
    heating_df_path = joinpath(@__DIR__, "measurements_heating_and_cooling.csv")
    heating_df = prepare_dataframe(heating_df_path)
end
```

```
[294.693, 294.693, 294.693, 294.693, 294.693, 294.693, 295.015, 295.015, 295.338, 295.338
```

```
begin
    # define data for the ODE
    y0 = Float64[294.70,]

    end_point = 1800
    tspan = (0.0, end_point)
    tsteps = range(tspan[1], tspan[2], length=60)

    # seconds since start
    t = heating_df[1:end_point, :time] .- heating_df.start_time[1]
    u = heating_df[1:end_point, :Q1]
    y = heating_df[1:end_point, :T1]
end
```

```
[294.693, 300.465, 310.423, 319.415, 326.51, 332.505, 337.116, 341.003, 344.227, 346.517,
```

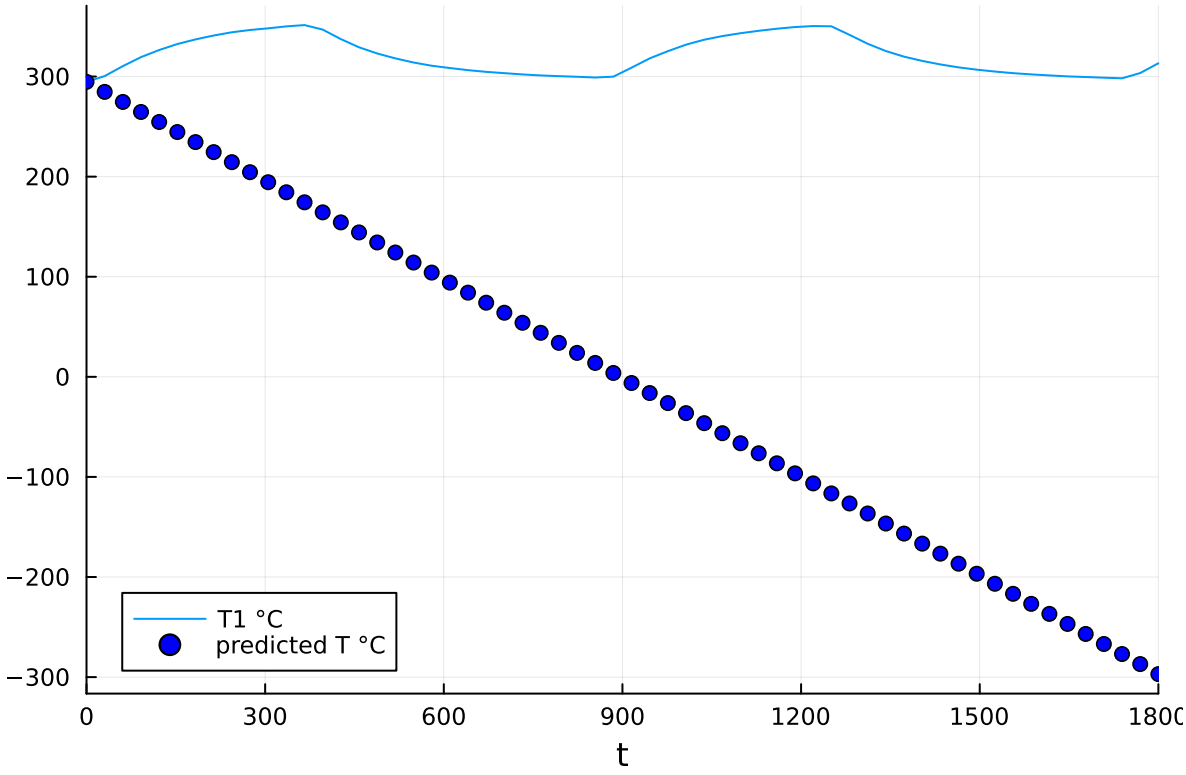
```
begin
    # Interpoliere the u, y
    u_t = LinearInterpolation(u, t)
    y_t = LinearInterpolation(y, t)

    ode_data = Array(y_t(tsteps))
end
```

```
loss_n_ode (generic function with 1 method)
```

```
begin
    # mgl. durch Interpolation so langsam im Training?
    dudt = Flux.Chain(x -> u_t.(x),  # wie Verallgemeinderung auf andere u_t
                      Flux.Dense(1=>16, tanh),
                      Flux.Dense(16=>1)) |> f64

    n_ode = NeuralODE(dudt, tspan, Tsit5(), saveat=tsteps, reltol=1e-7, abstol=1e-9)

    y_predict_before = n_ode(y0)
    ps = Flux.params(n_ode)

    function predict_n_ode()
          n_ode(y0)
    end

    loss_n_ode() = sum(abs2,ode_data .- transpose(predict_n_ode()))
end
```

```
begin
    opt_n_ode = ADAM(0.0001)
    data = Iterators.repeated((), 1000)  # training dauert sehr lange, daher
    interator r
    Flux.train!(loss_n_ode, ps, data, opt_n_ode)
end
```

100%




```
begin
    y_predict_after = n_ode(y0)

    # plot result
    p_n_ode = plot(tsteps, ode_data, label="T1 °C", mc=:red)
    # scatter!(p_n_ode, y_predict_before, label="before training T1 °C", mc=:green)
    scatter!(p_n_ode, y_predict_after, label="predicted T °C", mc=:blue)
end
```