Photo by Siora Photography (https://unsplash.com/@siora18) on Unsplash (https://unsplash.com)

# Units of measurement

🕒 2 minute read

In this lesson we will learn how to use units of measurement in Julia. We will learn how to convert from one unit to another and we will see how to write functions which can deal with units.

# `Unitful.jl`

The package used to deal with units of measurement in Julia is `Unitful.jl` . First of all we need to install `Unitful` :

```
using Pkg
Pkg.add("Unitful")
using Unitful
```

In order to add units of measurement to numbers we use the notation `u"unit"` :

```
one_meter = 1u"m"
```

or

```
one_meter = 1*u"m"
```

It is possible to convert from one unit to another using `uconvert` (https://painterqubits.github.io/Unitful.jl/stable/conversion/):

```
b = uconvert(u"km", one_meter)
>>>b
1//1000 km

>>>one_meter
1 m
```

As you can see on line 3 and 6, `uconvert` won't change the unit of the argument variable and will return a variable of the desired unit.

In case you want to remove the unit from a variable, you can use `ustrip(unit, variable)` : this will first convert the `variable` to the desired `unit` and then strip the unit.

```
c = ustrip(u"m", one_meter)

>>>c
1

>>>typeof(c)
Int64

>>>one_meter
1 m

>>>ustrip(u"km", one_meter)
1//1000
```

If no conversion is need, one can simply type:

```
>>>ustrip(one_meter)
1
```

# Functions

It is possible to write functions that accept arguments with units without any particular change:

```
function compute_speed(Δx, Δt)
    return Δx/Δt
end

>>>compute_speed(1u"km", 2u"s")
0.5 km s^-1
```

It is also possible to write functions with type annotations specific for arguments with units. `Unitful` provides many **abstract types** such as `Unitful.Length` or `Unitful.Time` , which are useful for type annotation of function arguments:

```
function compute_speed(Δx::Unitful.Length, Δt::Unitful.Time)
    return uconvert(u"m/s", Δx/Δt)
end

>>>compute_speed(1u"km", 2u"s")
500.0 m s^-1
```

Although it may be tempting, it is better to **refrain from using abstract types inside type definitions**. When defining a `struct`, you can use `typeof` to get the right type for the annotations:

```
struct Person
    height::typeof(1.0u"m")
    mass::typeof(1.0u"kg")
end
```

# Numerical integration

In Julia it is possible to compute integrals numerically taking into account units of measurements. For example, if we integrate a velocity over an interval of time we will get the distance covered:

```
1 using QuadGK
2 velocity(t::Unitful.Time) = 2u"m/s^2"*t + 1u"m/s"
3
4 >>>quadgk(velocity, 0u"s", 3u"s")[1]
5 12.0 m
```

For a uniformly accelerated motion, the velocity (line 2) is given by:

$$v(t) = at + v_0$$

Here `a = 2 m/s^2`.

If we integrate the velocity from `t = 0s` to `t = 3s` we get the distance covered, which is `12m` in this case. Notice how `QuadGK` automatically returns the correct unit (meters).

# Conclusions

In this lesson we have learnt how to add units of measurement using `Unitful.jl` and how to write functions which naturally use units. Furthermore, we have seen how `QuadGK` handles correctly units inside integrals.

If you liked this lesson and you would like to receive further updates on what is being published on this website, I encourage you to subscribe to the **newsletter** (https://techytok.com/newsletter/)! If you have any **question** or **suggestion**, please post them in the **discussion below**!

Thank you for reading this lesson and see you soon on TechyTok!

**Tags:** Julia

**Categories:** Lessons Tutorial

**Updated:** December 26, 2019

## LEAVE A COMMENT

ALSO ON TECHYTOK

Control Flow

8 months ago • 5 comments

From zero to Julia Lesson 4. Control Flow

Multiprocessing in Julia: writing a module

9 months ago • 3 comments

A tutorial on how to write a module in Julia which uses multiprocessing

Parallel computin

7 months ago • 4 comm

From zero to Julia Le 15. Parallel computin

0 Comments techytok 🔒 Privacy Policy

1 Login

♡ Recommend Tweet Share Sort by Best

Start the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

Be the first to comment.

✉ Subscribe Add Disqus to your siteAdd DisqusAdd ⚠ Do Not Sell My Data