Units of Measure

✗ 42 m/s + 237.8 ft/s

✓ 45 kg * 10 m/s^2 == 450 N

Accidents

[illegible]

[illegible]

F#

[illegible]

[illegible]

Declaration

[illegible]

Usage

[illegible]

Generics

[illegible]

Parametrized Types

[illegible]

C++ - Boost.Unit

Template Metaprogramming

Operator Overloading

No Runtime Overhead

Concepts

[illegible]

Unit System Agnostic

[illegible]

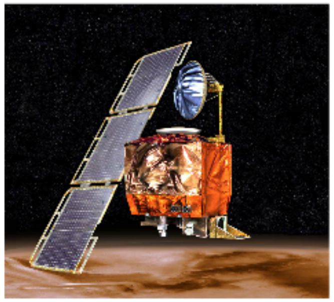
Usage

[illegible]

Units of Measure

- ✗ 42 m/s + 137.8 ft/s
- ✓ 45 kg * 10 m/s^2 == 450 N

Accidents



Mars Climate Orbiter

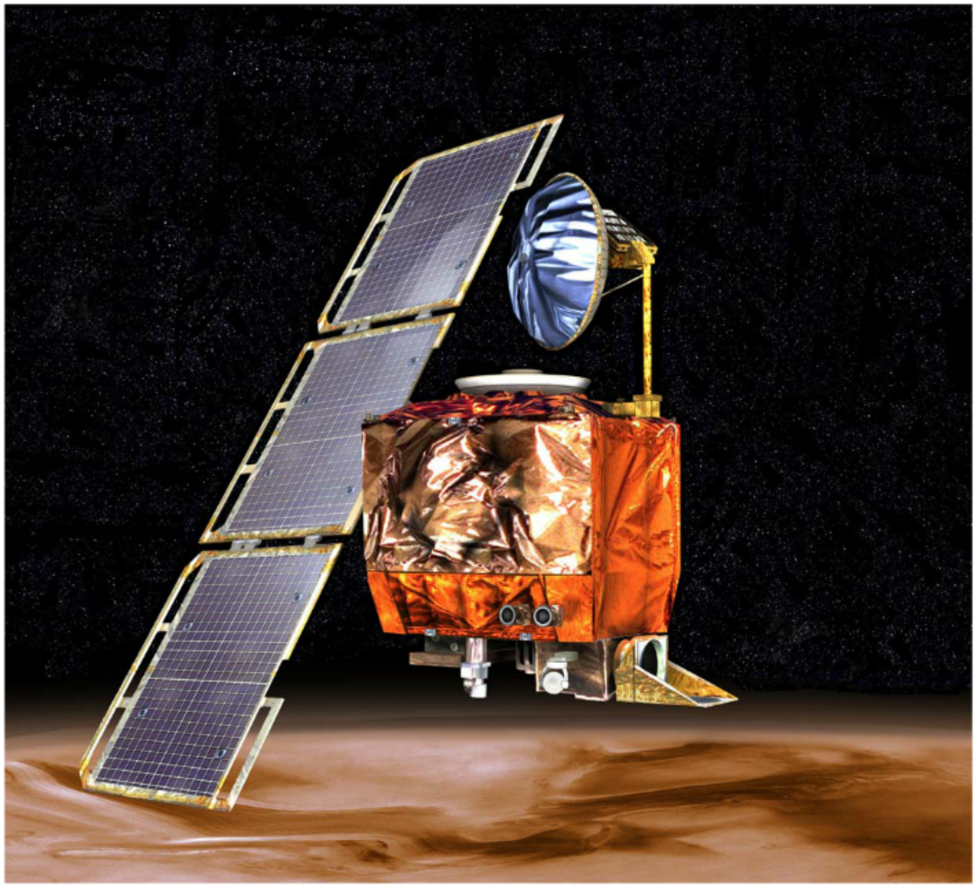
lbs·s <-> N·s

Units of Measure

✗ 42 m/s + 137.8 ft/s

✓ 45 kg * 10 m/s^2 == 450 N

Accidents



Mars Climate Orbiter

lbs·s <-> N·s

F#

compile time only

unit inference

Declaration

user defined units

```
[<Measure>] type m
[<Measure>] type s

[<Measure>] type N = kg m/s^2
```

Usage

```
let gravity = 9.81<m/s^2>    // acceleration
let windowHeight = 5.6 m

let impactSpeed = sqrt(2 * gravity * windowHeight)
// impactSpeed: float<m/s>
```

Generics

```
let sqr (x:float<_>) = x*x
// sqr: float<'u> -> float<'u^2>

let variance xs = let m = mean xs
    in mean_by (fun x -> sqr(x-m)) xs
// variance: float<'u> list -> float<'u^2>
```

Parametrized Types

```
type Vector2< [<Measure>] 'u> =
    { x: float<'u>; y: float<'u> }


let center = { x = 4.0<m>; y = 3.2<_> }
// center: Vector2<m>
```

F#

compile time only

unit inference

Declaration

user defined units

```
[<Measure>] type m
[<Measure>] type s

[<Measure>] type N = kg m/s^2
```

Usage

```
let gravity = 9.81<m/s^2>     // acceleration
let windowHeight = 5.6 m

let impactSpeed = sqrt(2 * gravity * windowHeight)
//  impactSpeed: float<m/s>
```

Generics

```
let sqr (x:float<_>) = x*x
//  sqr: float<'u>  ->  float<'u^2>

let variance xs = let m = mean xs
        in mean_by (fun x -> sqr(x-m)) xs
//  variance: float<'u> list  -> float<'u^2>
```

Parametrized Types

```
type Vector2< [<Measure>] 'u> =
    { x: float<'u>;  y: float<'u> }


let center = { x = 4.0<m>;  y = 3.2<_> }
//  center: Vector2<m>
```

C++ - BoosT.Unit

Template Metaprogramming

Operator Overloading

No Runtime Overhead

Concepts

Base Dimension	length, time
Base Unit	meter, second
Unit	m^2 * s
System	SI-System
Quantity	42 m/s

Usage

```
quantity<length, double>  dx(2.0 * si::meter);

quantity<length, measurement<double> >
  u(measurement<double>(4.52, 0.02)*meters);
```

Unit System Agnostic

Homogeneous vs Heterogeneous Systems

```
template<class System, class Y>
quantity<unit<energy_dimension, System>,Y>
work(quantity<unit<force_dimension,  System>, Y> F,
       quantity<unit<length_dimension,System>, Y> dx)
{ return F*dx; }
```

C++ - Boost.Unit

Template Metaprogramming

Operator Overloading

No Runtime Overhead

Concepts

Base Dimension	length, time
Base Unit	meter, second
Unit	m^2 * s
System	SI-System
Quantity	42 m/s

Usage

```
quantity<length, double>  dx(2.0 * si::meter);

quantity<length, measurement<double> >
   u(measurement<double>(4.52, 0.02)*meters);
```

Unit System Agnostic

Homogeneous vs Heterogeneous Systems

```
template<class System, class Y>
quantity<unit<energy_dimension, System>,Y>
work(quantity<unit<force_dimension,   System>, Y> F,
        quantity<unit<length_dimension,System>, Y> dx)
{  return F*dx;  }
```