Error Handling

Adám Brudzewsky

Error Handling

Adám Brudzewsky

Overview

Overview

- Event

Overview

Event

- Error

Overview

Event

- Error
- Interrupt

Overview

- Event
  - Error

Overview

Event

- Error

Event Number

Overview

- Event
  - Error
- Event Number
- Event Message

Overview

Error

Error Number

Error Message

Overview

Error

Error Number

Error Message

Trapping an Error

Overview

Error

Error Number

Error Message

Trapping an Error

- Tradfn

Overview

Error

Error Number

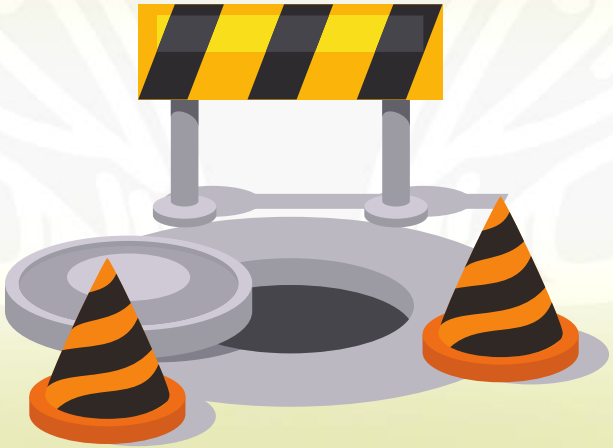
Error Message

Trapping an Error

- Tradfn
- Dfn

Error

normal execution cannot continue



Error

```
'yes'[4]
```

Error

```
'yes'[4]
```

Error

```
1 2 3+4 5
```

Error

```
1 2 3+4 5
    ↑   ↑
```

Error

```
4÷0
```

Error

```
4÷0
  ↑↑
```

Error Number

identity of a specific error

Error Number

```
'yes'[4]
```

Error Number

```
      'yes'[4]
INDEX ERROR
      'yes'[4]
           ^
```

Error Number

```
      'yes'[4]
INDEX ERROR
      'yes'[4]
           ^
      ⎕EN
```

Error Number

```
      'yes'[4]
INDEX ERROR
      'yes'[4]
           ^
      ⎕EN
3
```

Error Number

Error Number

```
1 2 3+4 5
```

Error Number

```
      1 2 3+4 5
LENGTH ERROR: Mismatched left and right argument shapes
      1 2 3+4 5
          ^
```

Error Number

```
      1 2 3+4 5
LENGTH ERROR: Mismatched left and right argument shapes
      1 2 3+4 5
            ^
      ⎕EN
5
```

Error Number

Error Number

```
      4÷0
DOMAIN ERROR: Divide by zero
      4÷0
       ^
```

Error Number

```
      4÷0
DOMAIN ERROR: Divide by zero
      4÷0
        ^
      ⎕EN
11
```

Error Message

name for a specific error number

Error Message

```
      'yes'[4]
INDEX ERROR
      'yes'[4]
           ^
      ⎕EN
3
```

Error Message

```
      'yes'[4]
INDEX ERROR
      'yes'[4]
           ^
      ⎕EN
3
      ⎕EM 3
```

Error Message

```
      'yes'[4]
INDEX ERROR
      'yes'[4]
           ^
      ⎕EN
3
      ⎕EM 3
INDEX ERROR
```

Error Message

```
      'yes'[4]
INDEX ERROR
      'yes'[4]
           ^
      ⎕EN
3
      ⎕EM 3
INDEX ERROR
```

Error Message

```
      'yes'[4]
INDEX ERROR
      'yes'[4]
           ^
      ⎕EN
3
      ⎕EM 3
INDEX ERROR
```

Error Message

```
      4 3 5+1 2
LENGTH ERROR: Mismatched left and right argument shapes
      4 3 5+1 2
            ^
      ⎕EN
5
```

Error Message

```
      4 3 5+1 2
LENGTH ERROR: Mismatched left and right argument shapes
      4 3 5+1 2
            ^
      ⎕EN
5
      ⎕EM 5
```

Error Message

```
      4 3 5+1 2
LENGTH ERROR: Mismatched left and right argument shapes
      4 3 5+1 2
            ^
      ⎕EN
5
      ⎕EM 5
LENGTH ERROR
```

Error Message

```
      4 3 5+1 2
LENGTH ERROR: Mismatched left and right argument shapes
      4 3 5+1 2
            ^
      ⎕EN
5
      ⎕EM 5
LENGTH ERROR
```

Error Message

```
      4 3 5÷0
DOMAIN ERROR: Divide by zero
      4 3 5÷0
            ^
      ⎕EN
11
```

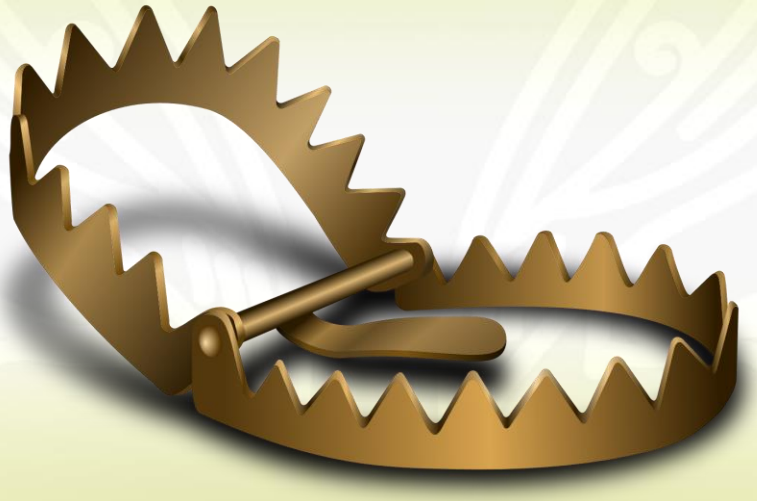
Error Message

```
      4 3 5÷0
DOMAIN ERROR: Divide by zero
      4 3 5÷0
            ^
      ⎕EN
11
      ⎕EM 11
DOMAIN ERROR
```

Error Message

```
      4 3 5÷0
DOMAIN ERROR: Divide by zero
      4 3 5÷0
          ^
      ⎕EN
11
      ⎕EM 11
DOMAIN ERROR
```

Trapping an Error



Trapping an Error

Method	Where	Syntax
structure	tradfns	`:Trap`
error guard	dfns	`::`
global state	both	`⎕TRAP`

Trapping an Error

Method	Where	Syntax
structure	tradfns	`:Trap`
error guard	dfns	`::`

Trapping an Error

tradfn structure

```
:Trap
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :If y
      z←'yes'
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

Trapping an Error: :Trap structure

```
∇ z←Foo y                    Foo 1
  :If y
      z←'yes'
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :If y
      z←'yes'
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1
yes
done
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :If y
      z←'yes'
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1
yes
done

      Foo 0
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :If y
      z←'yes'
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1
yes
done
      Foo 0
oh no
done
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :If y
      z←'yes'
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11
      z←'yes'
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
Foo 1
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1
4 3 5 0
d o n e
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1
4 3 5 0
d o n e
      Foo 0
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1
4 3 5 0
d o n e
      Foo 0
oh no
done
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
Foo 1 2
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1 2
LENGTH ERROR: Mismatched
Foo[2] ⎕←4 3 5÷y
                ^
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11 5
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11 5
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
Foo 1 2
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11 5
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1 2
oh no
done
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11 5
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1 2
oh no
done
      Foo 0
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11 5
      z←4 3 5÷y
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1 2
oh no
done
      Foo 0
oh no
done
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :If y
      z←'yes'
  :Else
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1
yes
done
      Foo 0
oh no
done
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Select y
  :Case 1
      z←'yes'
  :Case 0
      z←'oh no'
  :End
  z←↑z'done'
∇
```

```
      Foo 1
yes
done
      Foo 0
oh no
done
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 11 5
      z←4 3 5÷y
  :Case 11
      z←'bad!'
  :Case 5
      z←'count!'
  :End
  z←↑z'done'
```

```
      Foo 1 2
count!
done
      Foo 0
bad!
done
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 0
      z←4 3 5÷y
  :Case 11
      z←'bad!'
  :Case 5
      z←'count!'
  :End
  z←↑z'done'
```

```
      Foo 1 2
count!
done
      Foo 0
bad!
done
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 0
      z←4 3 5÷y
  :Case 11
      z←'bad!'
  :Case 5
      z←'count!'
  :Else
      z←'oh no'
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 0
      z←4 3 5÷y
  :Case 11
      z←'bad!'
  :Case 5
      z←'count!'
  :Else
      z←'oh no'
```

```
      Foo¨(1 2) 0
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 0
      z←4 3 5÷y
  :Case 11
      z←'bad!'
  :Case 5
      z←'count!'
  :Else
      z←'oh no'
```

```
      Foo¨(1 2) 0
count!  bad!
done    done
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 0
      z←4 3 5÷y
  :Case 11
      z←'bad!'
  :Case 5
      z←'count!'
  :Else
      z←'oh no'
```

```
      Foo¨(1 2) 0
count!  bad!
done    done
      Foo 1 2⍴3
```

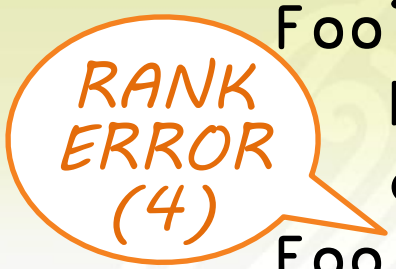
Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 0
      z←4 3 5÷y
  :Case 11
      z←'bad!'
  :Case 5
      z←'count!'
  :Else
      z←'oh no'
```

```
      Foo¨(1 2) 0
 count!  bad!
 done    done
      Foo 1 2⍴3
oh no
done
```

Trapping an Error: :Trap structure

```
∇ z←Foo y
  :Trap 0
      z←4 3 5÷y
  :Case 11
      z←'bad!'
  :Case 5
      z←'count!'
  :Else
      z←'oh no'
```




```
      Foo¨(1 2) 0
bad!
done
      Foo 1 2⍴3
oh no
done
```

Messages instead of Numbers

Messages instead of Numbers

- Simple variables
- Errors' namespace
- ⎕EM inverse function

Messages instead of Numbers: variables

```
(INDEX RANK LENGTH VALUE)←3 4 5 6
```

Messages instead of Numbers: variables

```
(INDEX RANK LENGTH VALUE)←3 4 5 6

:Trap RANK LENGTH
```

Messages instead of Numbers: namespace

```
ERROR←⎕NS⍬
```

Messages instead of Numbers: namespace

```
ERROR←⎕NS⍬
names←' ERROR' '\W+'⎕R'' '_' ⎕EM ⍳99
```

Messages instead of Numbers: namespace

```
ERROR←⎕NS⍬
names←' ERROR' '\W+'⎕R'' '_' ⎕EM ⍳99
ERROR⍎⍕'(',names,')←',⍳99
```

Messages instead of Numbers: namespace

```
ERROR←⎕NS⍬
names←' ERROR' '\W+'⎕R'' '_' ⎕EM ⍳99
ERROR⍎⍕'(',names,')←',⍳99

:Trap ERROR.DOMAIN
```

Messages instead of Numbers: namespace

```
ERROR←⎕NS⍬
names←' ERROR' '\W+'⎕R'' '_' ⎕EM ⍳99
ERROR⍎⍕'(',names,')←',⍳99

:Trap ERROR.DOMAIN
:Trap ERROR.(RANK LENGTH)
```

Messages instead of Numbers: function

```
ERROR←(' ERROR'⎕R''⎕EM⍳99)⍳⊆
```

Messages instead of Numbers: function

```
ERROR←(' ERROR'⎕R''⎕EM⍳99)⍳⊆

:Trap ERROR 'RANK' 'LENGTH'
```

Trapping an Error

dfn error guard

: :

Trapping an Error: :: error guard

```
Foo←{
   'yes'[⍵]
}
```

Trapping an Error: :: error guard

```
Foo←{
   'yes'[⍵]
}
```

```
      Foo 4
INDEX ERROR
Foo[1] 'yes'[⍵]
               ^
```

Trapping an Error: :: error guard

```
Foo←{
   3::'oh no'
   'yes'[⍵]
}
```

Trapping an Error: :: error guard

```
Foo←{
   3::'oh no'
   'yes'[⍵]
}
```

```
      Foo 3
2
      Foo 4
oh no
```

Trapping an Error: :: error guard

```
Foo←{
    3 11::'oh no'
    'yes'[⍵]
}
```

Trapping an Error: :: error guard

```
Foo←{
   3 11::'oh no'
   'yes'[⍵]
}
```

```
      Foo '4'
oh no
      Foo 4
oh no
```

Trapping an Error: :: error guard

```
Foo←{
   0::'oh no'
   'yes'[⍵]
}
```

```
      Foo '4'
oh no
      Foo 4
oh no
```

Trapping an Error: :: error guard

```
Foo←{
   11::'bad!'
   5::'count!'
   4 3 5÷⍵
}
```

```
      Foo 1 2
count!
      Foo 0
bad!
```

Trapping an Error: :: error guard

```
Foo←{
   11::z←'bad!'
   5::z←'count!'
   z←4 3 5÷⍵
   ↑z'done'
}
```

```
      Foo 1 2
count!
done
      Foo 0
bad!
done
```

Trapping an Error: :: error guard

```
Foo←{
   11::z←'bad!'
   5::z←'count!'
   z←4 3 5÷⍵
   ↑z'done'
}
```

```
      Foo 1 2
count!
done

      Foo 0
bad!
done
```

Trapping an Error: :: error guard

```
Foo←{
   z←{
      11::'bad!'
      5::'count!'
      4 3 5÷⍵
   }⍵
   ↑z'done'
}
```

```
      Foo 1 2
count!
done
      Foo 0
bad!
done
```

Trapping an Error: :: error guard

```
Foo←{
   Bar←{
      11::'bad!'
      5::'count!'
      4 3 5÷⍵
   }
   ↑(Bar ⍵)'done'
}
```

```
      Foo 1 2
count!
done
      Foo 0
bad!
done
```

Error Handling – Part 1

Tradfn structure `:Trap errno` (0 means all)

Dfn error guard `errno::result_expression`

Error Number of last error `⎕EN`

Error Message for number `⎕EM`

Error Handling – Part 1

Number for given message
aplcart.info?q=:: message

Tradfn structure `:Trap errno` (0 means all)

Dfn error guard `errno::result_expression`

Error Number of last error `⎕EN`

Error Message for number `⎕EM`

Error Handling – Part 1

Number for given message
aplcart.info?q=:: message

Tradfn structure `:Trap errno` (0 means all)

Dfn error guard `errno::result_expression`

Err… N… …ror `⎕EN`

Press F1 *for full docs!* …er `⎕EM`

Upcoming webinars

Feb	25	BAA	open session
Mar	4	- - -	
Mar	11	BAA	open session
Mar	18	Dyalog	Boolean Scans and Reductions
Mar	25	BAA	open session
Apr	1	- - -	
Apr	8	BAA	open session
Apr	15	Dyalog	Error Handling – Part 2

Upcoming webinars

Feb	25	BAA	open session
Mar	4	- - -	
Mar	11	BAA	open session
Mar	18	Dyalog	Boolean Scans and Reductions
Mar	25	BAA	open session
Apr	1	- - -	
Apr	8	BAA	open session
Apr	15	Dyalog	Error Handling – Part 2

britishaplassociation.org