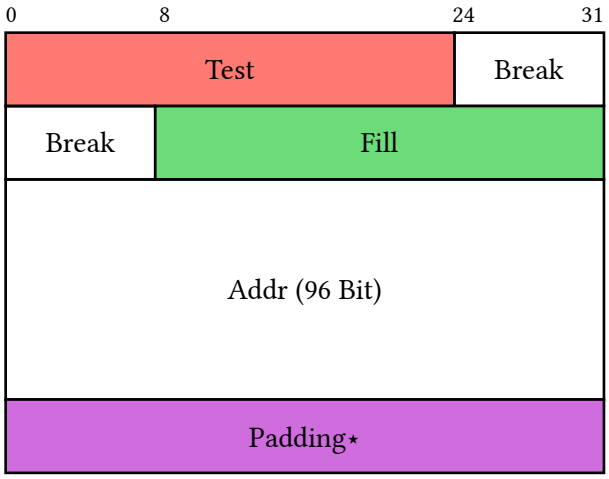
Bytefield

Colored Example

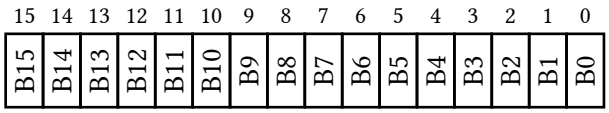
```
#bytefield(
  bytes(3,
    fill: red.lighten(30%)
  )[Test],
  bytes(2)[Break],
  bits(24,
    fill: green.lighten(30%)
  )[Fill],
  bytes(12)[Addr],
  padding(
    fill: purple.lighten(40%)
  )[Padding],
)
```



Show all bits in the bitheader

Show all bit headers with `bitheader: "all"`

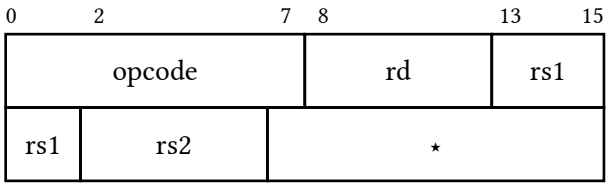
```
#bytefield(
    bits:16,
    msb_first: true,
    bitheader: "all",
    ..range(16).map(
      i => bit[#flagtext[B#i]]
    ).rev(),
)
```



Smart bit header

Show start and end bit of each bitbox with `bitheader: "smart"`.

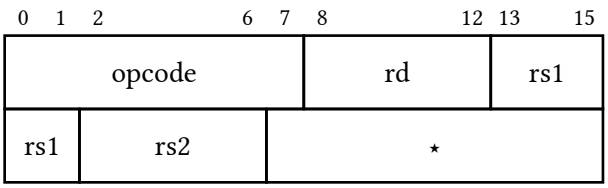
```
#bytefield(
  bits: 16,
  // same as
  // bitheader: (0,2,7,8,13,15),
  bitheader: "smart",
  bits(8)[opcode],
  bits(5)[rd],
  bits(5)[rs1],
  bits(5)[rs2],
  padding()[]
)
```



Bounds bit header

Show start bit of each bitbox with `bitheader: "bounds"`.

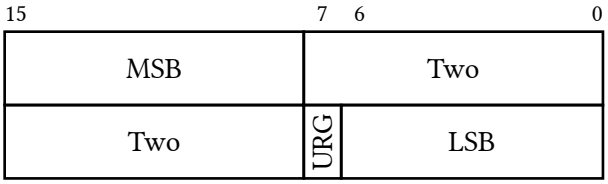
```
#bytefield(
  bits: 16,
  bitheader: "bounds",
  bits(8)[opcode],
  bits(5)[rd],
  bits(5)[rs1],
  bits(5)[rs1],
  bits(5)[rs2],
  padding()[]
)
```



Reversed bit order

Select `msb_first: true` for a reversed bit order.

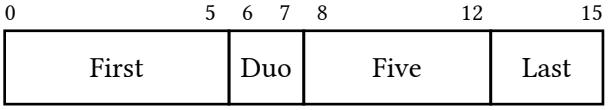
```
#bytefield(
    bits: 16,
    msb_first: true,
    bitheader: "smart",
    byte[MSB],
    bytes(2)[Two],
    bit[#flagtext("URG")],
    bits(7)[LSB],
)
```



Custom bit header

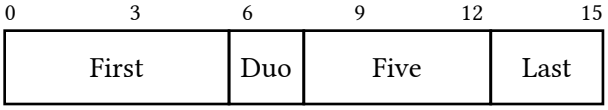
Pass an `array` to specify each number.

```
#bytefield(
    bits:16,
    bitheader: (0,5,6,7,8,12,15),
    bits(6)[First],
    bits(2)[Duo],
    bits(5)[Five],
    bits(3)[Last],
)
```



Pass an `integer` to show all multiples of this number.

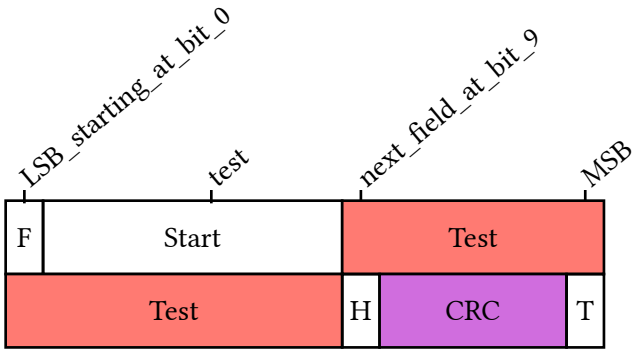
```
#bytefield(
    bits:16,
    bitheader: 3,
    bits(6)[First],
    bits(2)[Duo],
    bits(5)[Five],
    bits(3)[Last],
)
```



Text header instead of numbers [WIP]

Pass an `dictionary` as bitheader. Example:

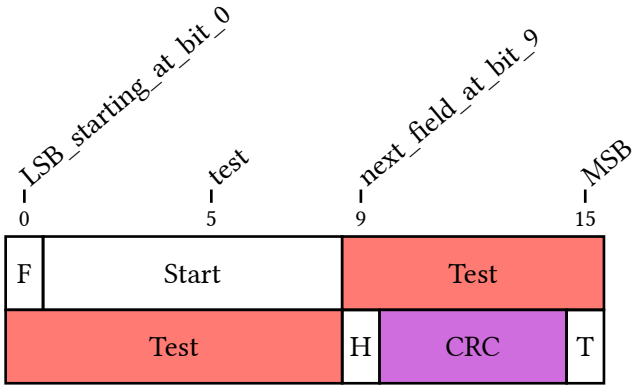
```
#bytefield(
  bitheader: (
    "0": "LSB_starting_at_bit_0",
    "5": "test",
    "9": "next_field_at_bit_9",
    "15":"MSB",
    angle: -40deg,
    marker: auto // or none
  ),
  bits: 16,
  bit[F],
  byte[Start],
  bytes(2,
    fill: red.lighten(30%)
  )[Test],
  bit[H],
  bits(5,
    fill: purple.lighten(40%)
  )[CRC],
  bit[T],
)
```



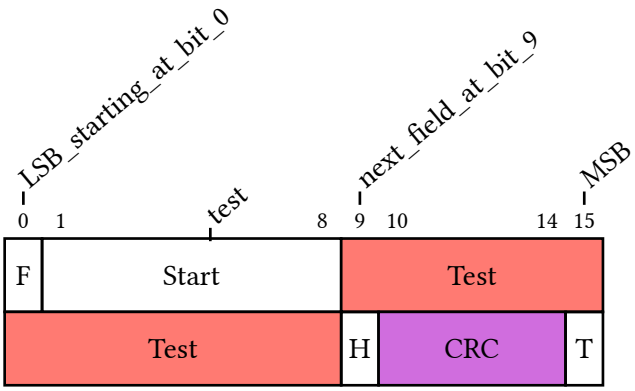
Text header and numbers [WIP]

You can also show labels and indexes by specifying `numbers`. `numbers` accepts the same string arguments as `bitheader`. You may also specify an array of indexes to show or simply `true` to show the index for each specified label.

```
#bytefield(
  bitheader: (
    "0": "LSB_starting_at_bit_0",
    "5": "test",
    "9": "next_field_at_bit_9",
    "15":"MSB",
    numbers: true,
    angle: -40deg,
    marker: auto // or none
  ),
  bits: 16,
  bit[F],
  byte[Start],
  bytes(2,
    fill: red.lighten(30%)
  )[Test],
  bit[H],
  bits(5,
    fill: purple.lighten(40%)
  )[CRC],
  bit[T],
)
```



```
#bytefield(
  bitheader: (
    "0": "LSB_starting_at_bit_0",
    "5": "test",
    "9": "next_field_at_bit_9",
    "15":"MSB",
    numbers: "bounds",
    angle: -40deg,
    marker: auto // or none
  ),
  bits: 16,
  bit[F],
  byte[Start],
  bytes(2,
    fill: red.lighten(30%)
  )[Test],
  bit[H],
  bits(5,
    fill: purple.lighten(40%)
  )[CRC],
  bit[T],
)
```

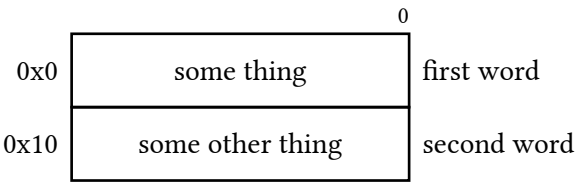


Pre/Post columns

Define additional columns with before the bitfield with `pre` or behind the bitfield with `post` and pass any tablex object.

You can use the helpers `left_aligned` and `right_aligned` for left and right aligned text.

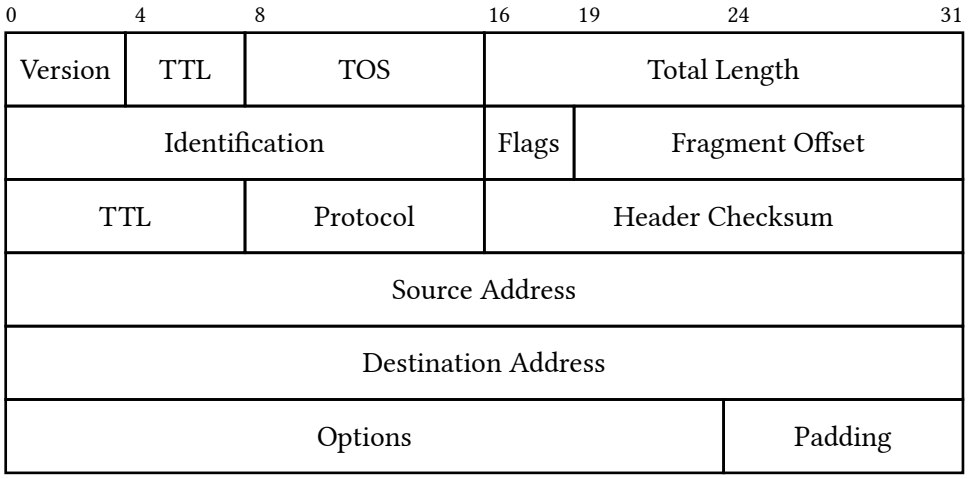
```
#bytefield(
  bits:1,
  pre:(auto,),
  post:(auto,),
  right_aligned[0x0],
  bit[some thing],
  left_aligned[first word],
  right_aligned[0x10],
  bit[some other thing],
  left_aligned[second word],
)
```



Some predefined network protocols

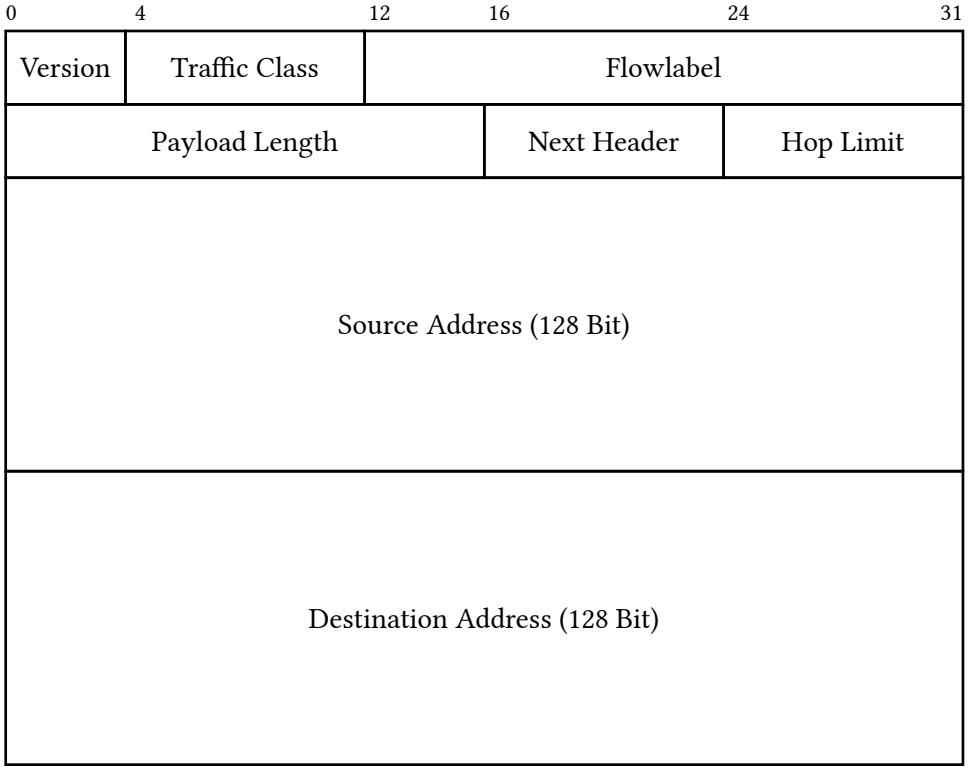
IPv4

```
#ipv4
```



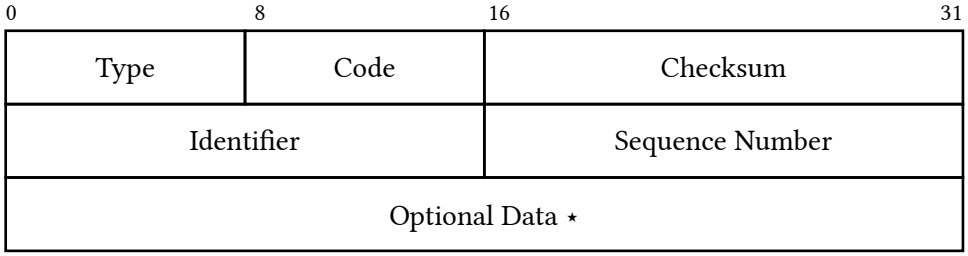
IPv6

```
#ipv6
```



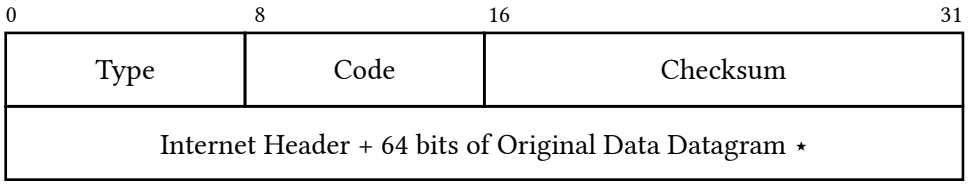
ICMP

```
#icmp
```



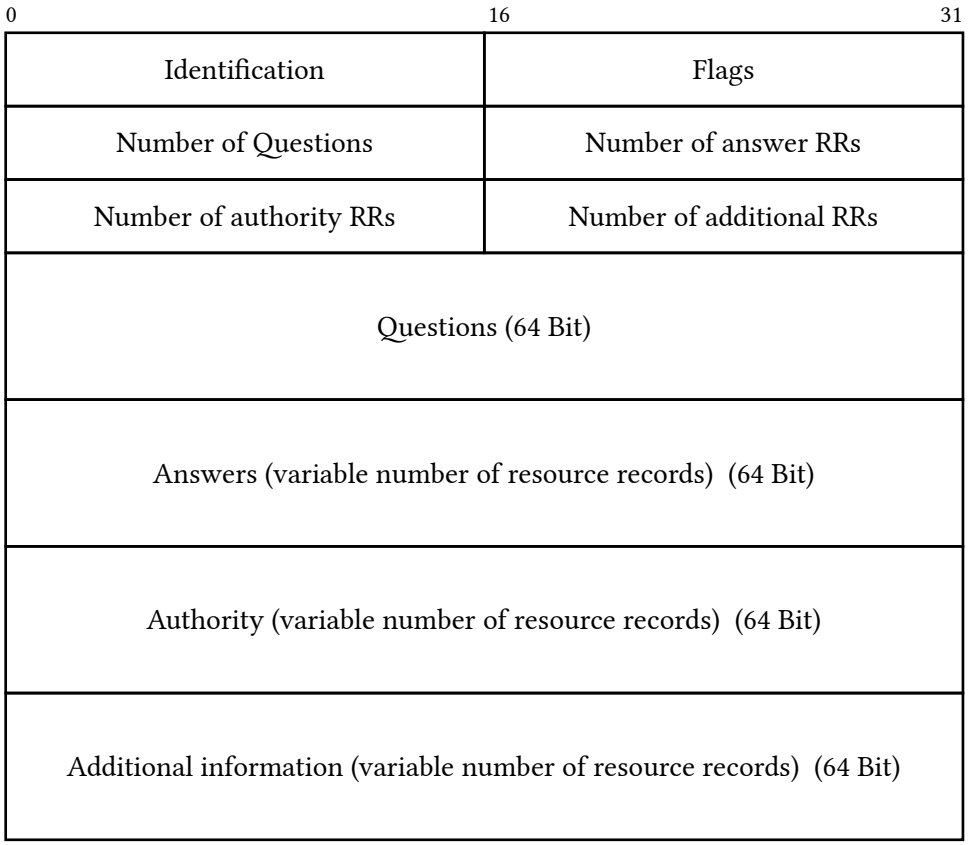
ICMPv6

```
#icmpv6
```



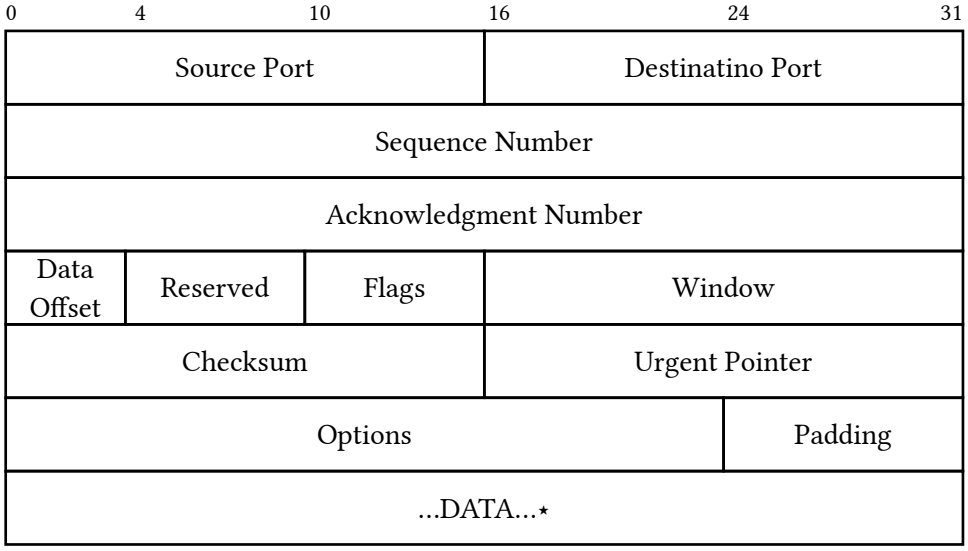
DNS

```
#dns
```

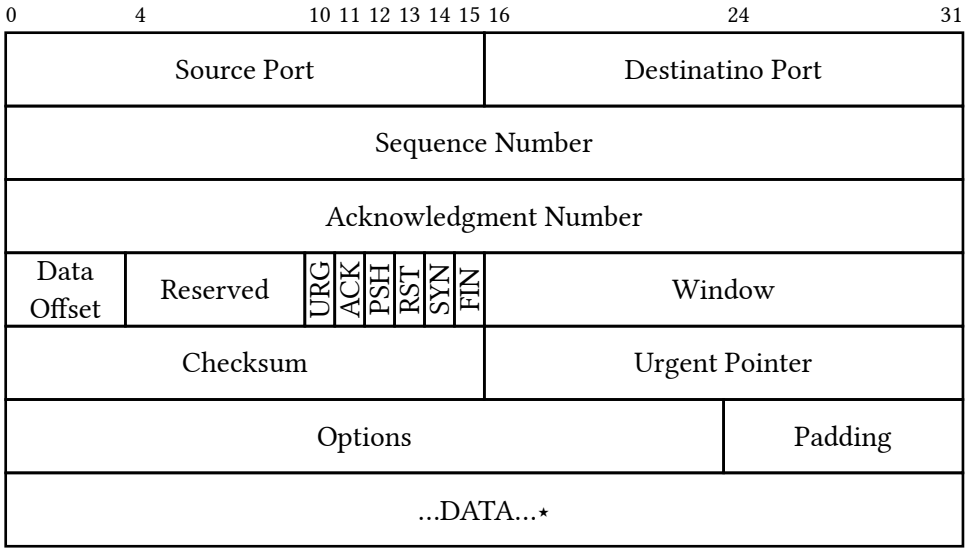


TCP

```
#tcp
```




```
#tcp_detailed
```



UDP

```
#udp
```