Trait std::str::FromStr

```
pub trait FromStr {
    type Err;
    fn from_str(s: &str) -> Result<Self, Self::Err>;
}
```

[−] Parse a value from a string

`FromStr`’s `from_str` method is often used implicitly, through `str`’s `parse` method. See `parse`’s documentation for examples.

`FromStr` does not have a lifetime parameter, and so you can only parse types that do not contain a lifetime parameter themselves. In other words, you can parse an `i32` with `FromStr`, but not a `&i32`. You can parse a struct that contains an `i32`, but not one that contains an `&i32`.

Examples

Basic implementation of `FromStr` on an example `Point` type:

Run

```
use std::str::FromStr;
use std::num::ParseIntError;

#[derive(Debug, PartialEq)]
struct Point {
    x: i32,
    y: i32
}

impl FromStr for Point {
    type Err = ParseIntError;

    fn from_str(s: &str) -> Result<Self, Self::Err> {
        let coords: Vec<&str> = s.trim_matches(|p| p == '(' || p == ')
                                 .split(',')
                                 .collect();

        let x_fromstr = coords[0].parse::<i32>()?;
        let y_fromstr = coords[1].parse::<i32>()?;

        Ok(Point { x: x_fromstr, y: y_fromstr })
    }
}

let p = Point::from_str("(1,2)");
assert_eq!(p.unwrap(), Point{ x: 1, y: 2} )
```

Associated Types

`type Err` [src]

The associated error which can be returned from parsing.

Required methods

`fn from_str(s: &str) -> Result<Self, Self::Err>` [src]

Parses a string `s` to return a value of this type.

If parsing succeeds, return the value inside `Ok`, otherwise when the string is ill-formatted return an error specific to the inside `Err`. The error type is specific to the implementation of the trait.

Examples

Basic usage with `i32`, a type that implements `FromStr`:

Run

```
use std::str::FromStr;

let s = "5";
let x = i32::from_str(s).unwrap();

assert_eq!(5, x);
```

Implementors

[+] `impl FromStr for IpAddr` 1.7.0 [src]

[+] `impl FromStr for SocketAddr` [src]

[+] `impl FromStr for bool` [src]

[+] `impl FromStr for char` 1.20.0 [src]

[+] `impl FromStr for f32` [src]

[+] `impl FromStr for f64` [src]

[+] `impl FromStr for i8` [src]

[+] `impl FromStr for i16` [src]

[+] `impl FromStr for i32` [src]

[+] `impl FromStr for i64` [src]

[+] `impl FromStr for i128` [src]

[+] `impl FromStr for isize` [src]

[+] `impl FromStr for u8` [src]

[+] `impl FromStr for u16` [src]

[+] `impl FromStr for u32` [src]

[+] `impl FromStr for u64` [src]

[+] `impl FromStr for u128` [src]

```
    mpl FromStr for usize                        [src]
[+] impl FromStr for OsString            1.45.0  [src]
[+] impl FromStr for Ipv4Addr                    [src]
[+] impl FromStr for Ipv6Addr                    [src]
[+] impl FromStr for SocketAddrV4         1.5.0  [src]
[+] impl FromStr for SocketAddrV6         1.5.0  [src]
[+] impl FromStr for NonZeroI8           1.35.0  [src]
[+] impl FromStr for NonZeroI16          1.35.0  [src]
[+] impl FromStr for NonZeroI32          1.35.0  [src]
[+] impl FromStr for NonZeroI64          1.35.0  [src]
[+] impl FromStr for NonZeroI128         1.35.0  [src]
[+] impl FromStr for NonZeroIsize        1.35.0  [src]
[+] impl FromStr for NonZeroU8           1.35.0  [src]
[+] impl FromStr for NonZeroU16          1.35.0  [src]
[+] impl FromStr for NonZeroU32          1.35.0  [src]
[+] impl FromStr for NonZeroU64          1.35.0  [src]
[+] impl FromStr for NonZeroU128         1.35.0  [src]
[+] impl FromStr for NonZeroUsize        1.35.0  [src]
[+] impl FromStr for PathBuf             1.32.0  [src]
[+] impl FromStr for String                      [src]
    impl FromStr for TokenStream
    impl FromStr for Literal
```