Trait Eq

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

Trait core::cmp::Eq

Trait for equality comparisons which are equivalence relations.

This means, that in addition to `a == b` and `a != b` being strict inverses, the equality must be (for all `a`, `b` and `c`):

- reflexive: `a == a`;
- symmetric: `a == b` implies `b == a`; and
- transitive: `a == b` and `b == c` implies `a == c`.

This property cannot be checked by the compiler, and therefore `Eq` implies `PartialEq`, and has no extra methods.

Derivable

This trait can be used with `#[derive]`. When `derive`d, because `Eq` has no extra methods, it is only informing the compiler that this is an equivalence relation rather than a partial equivalence relation. Note that the `derive` strategy requires all fields are `Eq`, which isn't always desired.

How can I implement `Eq`?

If you cannot use the `derive` strategy, specify that your type implements `Eq`, which has no methods:

```
enum BookFormat { Paperback, Hardback, Ebook }
struct Book {
    isbn: i32,
    format: BookFormat,
}
impl PartialEq for Book {
    fn eq(&self, other: &Book) -> bool {
        self.isbn == other.isbn
    }
}
impl Eq for Book {}
```

Run

Implementors

```
impl Eq for ParseFloatError
```
[src]

```
impl Eq for NonZeroU8
```
[src]

```
impl Eq for NonZeroU16
```
[src]

```
impl Eq for NonZeroU32
```
[src]

```
impl Eq for NonZeroU64
```
[src]

```
impl Eq for NonZeroU128
```
[src]

```
impl Eq for NonZeroUsize
```
[src]

```
impl Eq for NonZeroI8
```
[src]

```
impl Eq for NonZeroI16
```
[src]

```
impl Eq for NonZeroI32
```
[src]

```
impl Eq for NonZeroI64
```
[src]

```
impl Eq for NonZeroI128
```
[src]

```
impl Eq for NonZeroIsize
```
[src]

```
impl<T: Eq> Eq for Wrapping<T>
```
[src]

```
impl Eq for FpCategory
```
[src]

Trait Eq

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

impl Eq for ParseIntError [src]

impl<T> Eq for Discriminant<T> [src]

impl<T: Eq> Eq for ManuallyDrop<T> [src]

impl<T: Eq + Zeroable> Eq for NonZero<T> [src]

impl<T: ?Sized> Eq for *const T [src]

impl<T: ?Sized> Eq for *mut T [src]

impl<Ret> Eq for fn() -> Ret [src]

impl<Ret> Eq for extern "C" fn() -> Ret [src]

impl<Ret> Eq for unsafe fn() -> Ret [src]

impl<Ret> Eq for unsafe extern "C" fn() -> Ret [src]

impl<Ret, A> Eq for fn(_: A) -> Ret [src]

impl<Ret, A> Eq for extern "C" fn(_: A) -> Ret [src]

impl<Ret, A> Eq for extern "C" fn(_: A, ...) -> Ret [src]

impl<Ret, A> Eq for unsafe fn(_: A) -> Ret [src]

impl<Ret, A> Eq for unsafe extern "C" fn(_: A) -> Ret [src]

impl<Ret, A> Eq for unsafe extern "C" fn(_: A, ...) -> Ret [src]

impl<Ret, A, B> Eq for fn(_: A, _: B) -> Ret [src]

impl<Ret, A, B> Eq for extern "C" fn(_: A, _: B) -> Ret [src]

impl<Ret, A, B> Eq for extern "C" fn(_: A, _: B, ...) -> Ret [src]

impl<Ret, A, B> Eq for unsafe fn(_: A, _: B) -> Ret [src]

impl<Ret, A, B> Eq for unsafe extern "C" fn(_: A, _: B) -> Ret [src]

impl<Ret, A, B> Eq for unsafe extern "C" fn(_: A, _: B, ...) -> Ret [src]

impl<Ret, A, B, C> Eq for fn(_: A, _: B, _: C) -> Ret [src]

impl<Ret, A, B, C> Eq for extern "C" fn(_: A, _: B, _: C) -> Ret [src]

impl<Ret, A, B, C> Eq for extern "C" fn(_: A, _: B, _: C, ...) -> Ret [src]

impl<Ret, A, B, C> Eq for unsafe fn(_: A, _: B, _: C) -> Ret [src]

impl<Ret, A, B, C> Eq for unsafe extern "C" fn(_: A, _: B, _: C) -> Ret [src]

impl<Ret, A, B, C> Eq for unsafe extern "C" fn(_: A, _: B, _: C, ...) -> Ret [src]

impl<Ret, A, B, C, D> Eq for fn(_: A, _: B, _: C, _: D) -> Ret [src]

impl<Ret, A, B, C, D> Eq for extern "C" fn(_: A, _: B, _: C, _: D) -> Ret [src]

impl<Ret, A, B, C, D> Eq for extern "C" fn(_: A, _: B, _: C, _: D, ...) -> Ret [src]

impl<Ret, A, B, C, D> Eq for unsafe fn(_: A, _: B, _: C, _: D) -> Ret [src]

impl<Ret, A, B, C, D> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D) -> Ret [src]

impl<Ret, A, B, C, D> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, ...) -> Ret [src]

impl<Ret, A, B, C, D, E> Eq for fn(_: A, _: B, _: C, _: D, _: E) -> Ret [src]

impl<Ret, A, B, C, D, E> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E) -> Ret [src]

Trait Eq

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

impl<Ret, A, B, C, D, E> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, ...) -> Ret

impl<Ret, A, B, C, D, E> Eq for unsafe fn(_: A, _: B, _: C, _: D, _: E) -> Ret [src]

impl<Ret, A, B, C, D, E> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E) -> Ret [src]

impl<Ret, A, B, C, D, E> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F> Eq for fn(_: A, _: B, _: C, _: D, _: E, _: F) -> Ret [src]

impl<Ret, A, B, C, D, E, F> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F) -> Ret [src]

impl<Ret, A, B, C, D, E, F> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F> Eq for unsafe fn(_: A, _: B, _: C, _: D, _: E, _: F) -> Ret [src]

impl<Ret, A, B, C, D, E, F> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F) -> Ret [src]

impl<Ret, A, B, C, D, E, F> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G> Eq for fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G> Eq for unsafe fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H> Eq for fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H> Eq for unsafe fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I> Eq for fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I> Eq for unsafe fn(_: A, _: B, _: C, _: D, _: [src]

Trait Eq

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

E, _: F, _: G, _: H, _: I) -> Ret

impl<Ret, A, B, C, D, E, F, G, H, I> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J> Eq for fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J> Eq for unsafe fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K> Eq for fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K> Eq for unsafe fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> Eq for fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> Eq for extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L, ...) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> Eq for unsafe fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L) -> Ret [src]

impl<Ret, A, B, C, D, E, F, G, H, I, J, K, L> Eq for unsafe extern "C" fn(_: A, _: B, _: C, _: D, _: E, _: F, _: G, _: H, _: I, _: J, _: K, _: L, ...) -> Ret [src]

impl<T: ?Sized> Eq for NonNull<T> [src]

impl<T: ?Sized> Eq for PhantomData<T> [src]

impl<Y: Eq, R: Eq> Eq for GeneratorState<Y, R> [src]

impl Eq for RangeFull [src]

impl<Idx: Eq> Eq for Range<Idx> [src]

impl<Idx: Eq> Eq for RangeFrom<Idx> [src]

Trait Eq

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

```
impl<Idx: Eq> Eq for RangeTo<Idx>
```

```
impl<Idx: Eq> Eq for RangeInclusive<Idx>
```
[src]

```
impl<Idx: Eq> Eq for RangeToInclusive<Idx>
```
[src]

```
impl<T: Eq> Eq for Bound<T>
```
[src]

```
impl Eq for ()
```
[src]

```
impl Eq for bool
```
[src]

```
impl Eq for char
```
[src]

```
impl Eq for usize
```
[src]

```
impl Eq for u8
```
[src]

```
impl Eq for u16
```
[src]

```
impl Eq for u32
```
[src]

```
impl Eq for u64
```
[src]

```
impl Eq for u128
```
[src]

```
impl Eq for isize
```
[src]

```
impl Eq for i8
```
[src]

```
impl Eq for i16
```
[src]

```
impl Eq for i32
```
[src]

```
impl Eq for i64
```
[src]

```
impl Eq for i128
```
[src]

```
impl Eq for !
```
[src]

```
impl<'a, A: ?Sized> Eq for &'a A
where
    A: Eq,
```
[src]

```
impl<'a, A: ?Sized> Eq for &'a mut A
where
    A: Eq,
```
[src]

```
impl<T: Eq> Eq for Reverse<T>
```
[src]

```
impl Eq for Ordering
```
[src]

```
impl Eq for TypeId
```
[src]

```
impl<T: Eq> Eq for [T; 0]
```
[src]

```
impl<T: Eq> Eq for [T; 1]
```
[src]

```
impl<T: Eq> Eq for [T; 2]
```
[src]

```
impl<T: Eq> Eq for [T; 3]
```
[src]

```
impl<T: Eq> Eq for [T; 4]
```
[src]

```
impl<T: Eq> Eq for [T; 5]
```
[src]

```
impl<T: Eq> Eq for [T; 6]
```
[src]

```
impl<T: Eq> Eq for [T; 7]
```
[src]

```
impl<T: Eq> Eq for [T; 8]
```
[src]

```
impl<T: Eq> Eq for [T; 9]
```
[src]

```
impl<T: Eq> Eq for [T; 10]
```

Trait Eq

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

impl<T: Eq> Eq for [T; 11] [src]

impl<T: Eq> Eq for [T; 12] [src]

impl<T: Eq> Eq for [T; 13] [src]

impl<T: Eq> Eq for [T; 14] [src]

impl<T: Eq> Eq for [T; 15] [src]

impl<T: Eq> Eq for [T; 16] [src]

impl<T: Eq> Eq for [T; 17] [src]

impl<T: Eq> Eq for [T; 18] [src]

impl<T: Eq> Eq for [T; 19] [src]

impl<T: Eq> Eq for [T; 20] [src]

impl<T: Eq> Eq for [T; 21] [src]

impl<T: Eq> Eq for [T; 22] [src]

impl<T: Eq> Eq for [T; 23] [src]

impl<T: Eq> Eq for [T; 24] [src]

impl<T: Eq> Eq for [T; 25] [src]

impl<T: Eq> Eq for [T; 26] [src]

impl<T: Eq> Eq for [T; 27] [src]

impl<T: Eq> Eq for [T; 28] [src]

impl<T: Eq> Eq for [T; 29] [src]

impl<T: Eq> Eq for [T; 30] [src]

impl<T: Eq> Eq for [T; 31] [src]

impl<T: Eq> Eq for [T; 32] [src]

impl<T: Eq + Copy> Eq for Cell<T> [src]

impl<T: ?Sized + Eq> Eq for RefCell<T> [src]

impl Eq for ParseCharError [src]

impl Eq for CharTryFromError [src]

impl Eq for InvalidSequence [src]

impl<T: Eq> Eq for Option<T> [src]

impl Eq for NoneError [src]

impl<T: Eq, E: Eq> Eq for Result<T, E> [src]

impl<T: Eq> Eq for [T] [src]

impl Eq for SearchStep [src]

impl Eq for str [src]

impl Eq for ParseBoolError [src]

impl Eq for Utf8Error [src]

impl Eq for Error [src]

Trait Eq

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

impl Eq for Duration

impl Eq for Layout [src]

impl Eq for AllocErr [src]

impl Eq for CannotReallocInPlace [src]

impl Eq for CollectionAllocErr [src]

```
impl<A> Eq for (A,)
where
    A: Eq + ?Sized,
```
[src]

```
impl<A: Eq, B> Eq for (A, B)
where
    B: Eq + ?Sized,
```
[src]

```
impl<A: Eq, B: Eq, C> Eq for (A, B, C)
where
    C: Eq + ?Sized,
```
[src]

```
impl<A: Eq, B: Eq, C: Eq, D> Eq for (A, B, C, D)
where
    D: Eq + ?Sized,
```
[src]

```
impl<A: Eq, B: Eq, C: Eq, D: Eq, E> Eq for (A, B, C, D, E)
where
    E: Eq + ?Sized,
```
[src]

```
impl<A: Eq, B: Eq, C: Eq, D: Eq, E: Eq, F> Eq for (A, B, C, D, E, F)
where
    F: Eq + ?Sized,
```
[src]

```
impl<A: Eq, B: Eq, C: Eq, D: Eq, E: Eq, F: Eq, G> Eq for (A, B, C, D, E, F, G)
where
    G: Eq + ?Sized,
```
[src]

```
impl<A: Eq, B: Eq, C: Eq, D: Eq, E: Eq, F: Eq, G: Eq, H> Eq for (A, B, C, D, E,
F, G, H)
where
    H: Eq + ?Sized,
```
[src]

```
impl<A: Eq, B: Eq, C: Eq, D: Eq, E: Eq, F: Eq, G: Eq, H: Eq, I> Eq for (A, B, C,
D, E, F, G, H, I)
where
    I: Eq + ?Sized,
```
[src]

```
impl<A: Eq, B: Eq, C: Eq, D: Eq, E: Eq, F: Eq, G: Eq, H: Eq, I: Eq, J> Eq for (A,
B, C, D, E, F, G, H, I, J)
where
    J: Eq + ?Sized,
```
[src]

```
impl<A: Eq, B: Eq, C: Eq, D: Eq, E: Eq, F: Eq, G: Eq, H: Eq, I: Eq, J: Eq, K> Eq
for (A, B, C, D, E, F, G, H, I, J, K)
where
    K: Eq + ?Sized,
```
[src]

```
impl<A: Eq, B: Eq, C: Eq, D: Eq, E: Eq, F: Eq, G: Eq, H: Eq, I: Eq, J: Eq, K: Eq,
L> Eq for (A, B, C, D, E, F, G, H, I, J, K, L)
where
    L: Eq + ?Sized,
```
[src]

impl Eq for i8x2 [src]

impl Eq for u8x2 [src]

impl Eq for b8x2 [src]

impl Eq for i16x2 [src]

impl Eq for u16x2 [src]

impl Eq for i8x4 [src]

impl Eq for u8x4 [src]

impl Eq for b8x4 [src]

Trait Eq

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

impl Eq for i8x8 [src]

impl Eq for u8x8 [src]

impl Eq for b8x8 [src]

impl Eq for i16x4 [src]

impl Eq for u16x4 [src]

impl Eq for i32x2 [src]

impl Eq for u32x2 [src]

impl Eq for i8x16 [src]

impl Eq for u8x16 [src]

impl Eq for b8x16 [src]

impl Eq for i16x8 [src]

impl Eq for u16x8 [src]

impl Eq for i32x4 [src]

impl Eq for u32x4 [src]

impl Eq for i64x2 [src]

impl Eq for u64x2 [src]

impl Eq for i8x32 [src]

impl Eq for u8x32 [src]

impl Eq for b8x32 [src]

impl Eq for i16x16 [src]

impl Eq for u16x16 [src]

impl Eq for i32x8 [src]

impl Eq for u32x8 [src]

impl Eq for i64x4 [src]

impl Eq for u64x4 [src]

impl Eq for i8x64 [src]

impl Eq for u8x64 [src]

impl Eq for b8x64 [src]

impl Eq for i16x32 [src]

impl Eq for u16x32 [src]

impl Eq for i32x16 [src]

impl Eq for u32x16 [src]

impl Eq for i64x8 [src]

impl Eq for u64x8 [src]

impl Eq for CpuidResult [src]

impl<T: ?Sized + Eq> Eq for Box<T>

impl<T: ?Sized + Eq> Eq for Arc<T>

Trait Eq

Implementors

core::cmp

Structs

Reverse

Enums

Ordering

Traits

Eq
Ord
PartialEq
PartialOrd

Functions

max
min

```
impl<T: ?Sized + Eq> Eq for Rc<T>

impl<K: Eq, V: Eq> Eq for BTreeMap<K, V>

impl<T: Eq> Eq for BTreeSet<T>

impl<'a, B: ?Sized> Eq for Cow<'a, B>
where
    B: Eq + ToOwned,

impl<T: Eq> Eq for LinkedList<T>

impl Eq for String

impl Eq for ParseError

impl<T: Eq> Eq for Vec<T>

impl<A: Eq> Eq for VecDeque<A>

impl Eq for Span

impl Eq for LineColumn

impl Eq for SourceFile

impl Eq for Delimiter

impl Eq for Spacing

impl Eq for UnicodeVersion

impl Eq for DecodeUtf16Error

impl<'a> Eq for Utf8LossyChunk<'a>

impl Eq for TestName

impl Eq for NamePadding

impl Eq for BenchMode

impl Eq for ShouldPanic

impl Eq for TestDesc

impl Eq for OutputFormat
```