



Mojo traits

What each standard library trait does, when you need it, and where it bites

keyword type built-in "string" number operator @decorator True/False/None # comment

LIFECYCLE

AnyType	the root; every type conforms automatically	<code>builtin/anytype.mojo</code>
no requirements		

ImplicitlyDeletable	has an implicit destructor, called at last use	<code>builtin/anytype.mojo</code>
<code>__del__(deinit self, /)</code> provided		
<code>comptime __del__is_trivial: Bool</code>	compile-time flag	
Automatically added to every eligible type (all fields are also ImplicitlyDeletable). When the type is trivial, Mojo skips the destructor.		

Movable	transfer ownership; enables the ^ operator	<code>builtin/value.mojo</code>
<code>__init__(out self, *, deinit move: Self)</code> provided		
<code>comptime __move_ctor_is_trivial: Bool</code>	compile-time flag	
Required to store a type in List , Optional , or Variant , or to return it by move.		

Copyable	explicit copy	<code>builtin/value.mojo</code>
Refines: Movable		
<code>__init__(out self, *, copy: Self)</code> provided		
<code>copy(self) -> Self</code>	provided	
<code>comptime __copy_ctor_is_trivial: Bool</code>	compile-time flag	
The copy constructor is synthesized if all fields are Copyable .		

ImplicitlyCopyable	(MARKER) lets copies be inserted implicitly	<code>builtin/value.mojo</code>
Refines: Copyable < Movable · no new requirements		
Can mask logical errors and hide code reasoning. Prefer Movable or Copyable .		

Defaultable	create with no arguments	<code>builtin/value.mojo</code>
<code>__init__(out self)</code>		
Reach for this when you need parameterized default-construction without arguments.		

RegisterPassable	(MARKER) stored in registers, not memory; no stable address or identity	<code>builtin/value.mojo</code>
Refines: Movable · no requirements (marker)		
No stable address: you can't take the address of self in read-convention methods. Identifiable is meaningless for these types.		
Moved trivially; all fields must also conform.		

TrivialRegisterPassable	(MARKER) register-passable, copyable by moving bits, no side effects	<code>builtin/value.mojo</code>
Refines: ImplicitlyCopyable < Copyable < Movable, ImplicitlyDeletable, RegisterPassable · no requirements (marker)		
A type whose values are treated as basic bit patterns. No constructors or destructors needed. All fields must also conform.		

FORMAT

Writable	format itself as text; works with print(), String(), format strings	<code>format/__init___.mojo</code>
<code>write_to(self, mut writer: Some[Writer])</code> provided		
<code>write_repr_to(self, mut writer: Some[Writer])</code> provided		
If all fields conform, you inherit both methods through reflection.		

Writer	a destination for a custom output target	<code>format/__init___.mojo</code>
String, FileHandle, FileDescriptor conform		
<code>write_string(mut self, string: StringSlice)</code>		
<code>write[*Ts: Writable](mut self, *args: *Ts)</code> provided		
Use for loggers, network streams, string builders, etc.		

TESTING

Strategy	produces random inputs for property-based tests	<code>testing/prop/strategy/__init___.mojo</code>
Refines: ImplicitlyDeletable, Movable		
Value: Copyable & ImplicitlyDeletable	associated type	
<code>value(mut self, mut rng: Rng) raises -> Self.Value</code>		
Allows strategies to carry and advance state between draws. value() draws one sample from the random number generator.		

ACCELERATOR TRAITS

DevicePassable	marks a type as passable to an accelerator	<code>builtin/device_passable.mojo</code>
<code>comptime device_type: AnyType</code> the on-device type		
<code>_to_device_type(self, mut enc, ...)</code> DeviceContext hook		
A host type implements this so it can be handed to a GPU or other accelerator. DeviceContext calls the conversion hook to turn host into device at kernel launch.		

DeviceTypeEncoder	encodes host values into device layout	<code>builtin/device_passable.mojo</code>
<code>target() -> _TargetType</code> device target		
<code>encode_device_ptr(mut self, ...)</code> required		
<code>encode[T](mut self, value, dst)</code> provided (+ fields, tuple, array)		
Encodes a value's fields into the accelerator's data layout.		

COMPARE & HASH

Equatable	equality; enables == and !=	<code>builtin/comparable.mojo</code>
<code>__eq__(self, other: Self) -> Bool</code> provided		
<code>__ne__(self, other: Self) -> Bool</code> provided		
Don't use with floating-point values (use <code>isclose()</code>). NaN != NaN .		
Mojo provides a fieldwise default. Override for caches, internal metadata, and custom behavior.		

Comparable	ordered comparison; < > ≤ ≥, and sort()	<code>builtin/comparable.mojo</code>
Refines: Equatable		
<code>__lt__(self, rhs: Self) -> Bool</code>		
<code>__gt__(self, rhs: Self) -> Bool</code> provided		
<code>__le__(self, rhs: Self) -> Bool</code> provided		
<code>__ge__(self, rhs: Self) -> Bool</code> provided		
Implement <code>__lt__()</code> unless it's expensive. If so, override all four.		

Hashable	produces a hash; needed for Dict keys and Set elements	<code>hashlib/hash.mojo</code>
KeyElement = Hashable + Equatable + Movable		
<code>__hash__(self, mut hasher: Some[Hasher])</code> provided		

Hasher	implements a hash algorithm (the algorithm, not a hashable type)	<code>hashlib/hasher.mojo</code>
<code>__init__(out self)</code>		
<code>_update_with_bytes(mut self, data: Span[Byte, _])</code>		
<code>_update_with_simd(mut self, value: SIMD[_, _])</code>		
<code>update(mut self, value: Some[Hashable])</code>		
<code>finish(var self) -> UInt64</code>		
Hashers remain alive after finalization. All three update methods are required.		

Identifiable	identity; same-object test, enables is / is not	<code>builtin/identifiable.mojo</code>
<code>__is__(self, rhs: Self) -> Bool</code>		
<code>__isnot__(self, rhs: Self) -> Bool</code> provided		
Excludes register-passable types, which don't have stable addresses.		

CONVERT

Boolable, Intable, Floatable	convert with Bool(), Int(), Float64()	<code>builtin/bool.mojo · builtin/int.mojo · builtin/floatable.mojo</code>
<code>__bool__(self) -> Bool</code> Boolable		
<code>__int__(self) -> Int</code> Intable		
<code>__int__(self) raises -> Int</code> IntableRaising		
<code>__float__(self) -> Float64</code> Floatable		
<code>__float__(self) raises -> Float64</code> FloatableRaising		
If the method raises, use the Raising variant.		
Boolable unlocks if / while / and / or usage.		

MATH

Absable, Powable, Roundable	unary math operators	<code>math/math.mojo</code>
<code>__abs__(self) -> Self</code> Absable · abs()		
<code>__pow__(self, exp: Self) -> Self</code> Powable · pow(), **		
<code>__round__(self) -> Self</code> Roundable · round()		
<code>__round__(self, ndigits: Int) -> Self</code> Roundable · round(), precision		

Ceilable, Floorable, Truncable	round toward a bound	<code>math/math.mojo</code>
<code>__ceil__(self) -> Self</code> Ceilable · ceil()		
<code>__floor__(self) -> Self</code> Floorable · floor()		
<code>__trunc__(self) -> Self</code> Truncable · trunc()		

CeilDivable, CeilDivableRaising	ceiling division (rounds up instead of down)	<code>math/math.mojo</code>
<code>__ceildiv__(self, denominator: Self) -> Self</code> CeilDivable		
<code>__ceildiv__(self, denominator: Self) raises -> Self</code> CeilDivableRaising		

DivModable	combined division and modulo; enables divmod()	<code>math/math.mojo</code>
Refines: ImplicitlyCopyable < Copyable < Movable		
<code>__divmod__(self, denominator: Self) -> Tuple[Self, Self]</code>		
Math outlier. The tuple is (quotient, remainder).		

ITERATE

Sized, SizedRaising	has a length; enables len()	<code>builtin/len.mojo</code>
<code>__len__(self) -> Int</code> Sized		
<code>__len__(self) raises -> Int</code> SizedRaising		

Iterable	iterate by borrowing	<code>iter/__init___.mojo</code>
<code>IteratorType[iterable_mut: Bool, //, iterable_origin: Origin[mut=iterable_mut]]: Iterator</code> associated type		
<code>__iter__(ref self) -> Self.IteratorType[origin_of(self)]</code>		
Parameterized on mutability and origin. Yields references tied to the source lifetime.		

IterableOwned	iterate by consuming and owning	<code>iter/__init___.mojo</code>
<code>IteratorOwnedType: Iterator</code> associated type		
<code>__iter__(var self) -> Self.IteratorOwnedType</code>		
No origin tracking.		

Iterator	produces elements one at a time; the for-loop workhorse	<code>iter/__init___.mojo</code>
Refines: ImplicitlyDeletable, Movable		
Element: Movable associated type		
<code>__next__(mut self) raises StopIteration -> Self.Element</code>		
<code>bounds(self) -> Tuple[Int, Optional[Int]]</code> provided		
<code>nth(var self, n: Int) -> Optional[Self.Element]</code> provided		
Requires an Iterable on the collection, and Iterator on the iterator. Don't rely on bounds() for safety checks. It's a hint.		
Typed raises (StopIteration).		

INTEROP

PathLike	represents a file system path	<code>os/pathlike.mojo</code>
Path conforms		
<code>__fspath__(self) -> String</code>		

ConvertibleToPython	can be sent to Python	<code>python/conversions.mojo</code>
Refines: ImplicitlyDeletable		
<code>to_python_object(var self) raises -> PythonObject</code>		

ConvertibleFromPython	can be created from a Python object	<code>python/conversions.mojo</code>
Refines: Copyable < Movable, ImplicitlyDeletable		
<code>__init__(out self, *, py: PythonObject) raises</code>		