What is keywords

Keywords are predefined, reserved words used in programming that have special meanings to the compiler. Keywords are part of the syntax and they cannot be used as an identifier.

The following keywords currently have the functionality described.

- as - perform primitive casting, disambiguate the specific trait containing an item, or rename items in use and extern crate statements
- async - return a Future instead of blocking the current thread
- await - suspend execution until the result of a Future is ready
- break - exit a loop immediately
- const - define constant items or constant raw pointers
- continue - continue to the next loop iteration
- crate - link an external crate or a macro variable representing the crate in which the macro is defined
- dyn - dynamic dispatch to a trait object
- else - fallback for if and if let control flow constructs
- enum - define an enumeration
- extern - link an external crate, function, or variable
- false - Boolean false literal
- fn - define a function or the function pointer type
- for - loop over items from an iterator, implement a trait, or specify a higher-ranked lifetime
- if - branch based on the result of a conditional expression
- impl - implement inherent or trait functionality
- in - part of for loop syntax
- let - bind a variable
- loop - loop unconditionally
- match - match a value to patterns
- mod - define a module
- move - make a closure take ownership of all its captures
- mut - denote mutability in references, raw pointers, or pattern bindings
- pub - denote public visibility in struct fields, impl blocks, or modules
- ref - bind by reference
- return - return from function
- Self - a type alias for the type we are defining or implementing
- self - method subject or current module
- static - global variable or lifetime lasting the entire program execution
- struct - define a structure
- super - parent module of the current module
- trait - define a trait
- true - Boolean true literal

- type - define a type alias or associated type
- union - define a union and is only a keyword when used in a union declaration
- unsafe - denote unsafe code, functions, traits, or implementations
- use - bring symbols into scope
- where - denote clauses that constrain a type
- while - loop conditionally based on the result of an expression

Keyword reserved for future use

The following keywords do not have any functionality but are reserved by Rust for potential future use.

- abstract
- become
- box
- do
- final
- macro
- override
- priv
- try
- typeof
- unsized
- virtual
- yield