

## Index

- deducing
  - built-in arrays, 211–212
  - list initialization, 210–211
  - pointer types, 197–198
  - reference types, 198
- deeply nested variable types, 202–203
- default constructed, 478, 752
- default constructors, 754, 1136
  - declaring special member functions, 33–34
  - suppressed by `std::initializer_list`, 568–570
  - as trivial, 437
  - user-provided, 755
- default initialization, 216–219, 765
  - in aggregate initialization, 221
  - `constexpr` functions, 273
  - for `nonstatic data members`, 322–323
- default initialized, 493, 752
- default member initialization, 233
- default member initializers
  - aggregate initialization with, 138–141
  - annoyances, 328–330
  - applicability limitations, 329
  - array size deduction, lack of, 330
  - loss of aggregate status, 330
  - loss of triviality, 329–330
  - parenthesized direct-initialization syntax, lack of, 328–329
  - `constexpr` functions, 270
  - description of, 318–321
  - potential pitfalls, 326–328
    - inconsistent subobject initialization, 326–328
    - loss of insulation, 326
  - safety of, 6
  - trivial types, 426
  - union interactions, 320–321
  - use cases, 322–325
    - boilerplate repetition, avoiding, 323–325
    - documentation of default values, 325
    - `nonstatic data member` initialization, 322–323
    - simple struct initialization, 322
- default values, documentation of, 325
- defaulted default constructors, exception specifications and, 1087
- defaulted functions, 522, 649. *See also* deleted functions; rvalue references; `static_assert`
  - annoyances, 42–43
  - description of, 33–36
  - exception specifications and, 1086
  - first declaration of special member function, 34–35
  - further reading for, 44
  - implementation of user-provided special member function, 35–36
  - implicit generation of special member functions, 44–45
  - potential pitfalls, 41–42
  - use cases, 36–41
    - making explicit class APIs with no runtime cost, 38–39
    - physically decoupling interface from implementation, 40–41
    - preserving type triviality, 39–40
    - restoring generation of suppressed special member function, 36–37
- defaulted special member functions. *See* defaulted functions
- defaulted template parameters, 31
- default/value, 215
- defect reports (DR), 432n10, 615n2, 722n8, 1086n2
- defensive checks, 468, 744
- defensive programming, 1024
- defined behavior, 1112–1113
- defining declarations, 729
- definition (of objects), 68
- definitions, 315, 879
- delaying return-type deduction, 1199–1200
- delegating constructors
  - description of, 46–48
  - potential pitfalls, 50–51
    - delegation cycles, 50–51
    - suboptimal factoring, 51
  - use cases, 48–50
- delegation cycles, 50–51
- deleted functions, 33–34, 757, 1086n2. *See also* defaulted functions; rvalue references
  - annoyances, 58–59
  - description of, 53
  - further reading for, 60
  - as trivial, 523
  - use cases, 53–57
    - hiding structural base class member functions, 56–57
    - preventing implicit conversion, 55–56
    - suppressing special member function generation, 53–55
- dependency. *See* data dependency
- dependent base classes. *See* inheriting constructors
- dependent types
  - generic lambdas, 981
  - inheriting constructors, 538
- [[deprecated]] attribute, 14
  - description of, 147–148
  - potential pitfalls, 150
  - use cases, 148–150