to C++11 and C++14. For the major compilers, applying this `defect report` was either standardizing already existing practice or quickly adopting the changes.[9]

See Also

- “Defaulted Functions” (§1.1, p. 33) explains how defaulted functions are used to implement functions that might otherwise have been suppressed by inherited constructors.
- “Delegating ~~Ctors~~” (§1.1, p. 46) discusses how delegating constructors are used to call one constructor from another from within the same user-defined type.
- “Deleted Functions” (§1.1, p. 53) describes how deleted functions can be used to exclude inherited constructors that are unwanted entirely.
- “**`override`**” (§1.1, p. 104) explains how **`override`** can be used to ensure that a member function intended to override a virtual function actually does so.
- “Default Member Init” (§2.1, p. 318) discusses how defaulted member initializers can be used to provide nondefault values for data members in derived classes that make use of inheriting constructors.
- “Forwarding References” (§2.1, p. 377) describes how forwarding references are used as an alternative (workaround) when access levels differ from those for base-class constructors.
- “Variadic Templates” (§2.1, p. 873) explains how variadic templates are used as an alternative (workaround) when access levels differ from those for base-class constructors.

[9]For example, GCC 7.0 (c. 2017) and later and Clang 4.0 (c. 2017) and later all have the modern behavior fully implemented regardless of which standard version is chosen when compiling.