Glossary

boilerplate code – significant sections of source code that occur (typically copied) in multiple places with little or no variation. `using` Aliases (136), Variable Templates (161), Default Member Init (322), `enum class` (333)

brace elision – the act, by a programmer, of omitting braces around the initialization of nested members of aggregate type during aggregate initialization of a compound aggregate. Aggregate Init '14 (140)

brace initialization – the initialization of an object using a braced-initializer list. Generalized PODs '11 (493), Range `for` (684), *Rvalue* References (752), Variadic Templates (926)

braced-initializer list – a possibly empty, comma-separated sequence of values between braces ({ and }) used as an object initializer. `initializer_list` (554)

byte – the fundamental storage unit in the C++ memory model, necessarily at least 8 bits and, on modern hardware, universally exactly 8 bits. Digit Separators (153)

C linkage – a form of linkage that allows for declarations and definitions written in C++ to match with callers and callees written and compiled in the C language. Generalized PODs '11 (403)

C++03 POD type – (1) a scalar type, (2) an aggregate type or `union` having no nonstatic data members that are not themselves C++03 POD types, or (3) an array of such objects. Generalized PODs '11 (414)

C++11 memory allocator – a potentially stateful (as of C++11) object that provides operations for allocating and deallocating memory dynamically. An allocator type appears as an optional template parameter in many standard containers and can be used to control how memory is allocated, how elements are constructed, and how they are referenced; see also C++17 `pmr` allocator. *Rvalue* References (763)

C++11 POD type – a type that is both a trivial type and a standard-layout type. C++20 removes the notion of a POD type as it is no longer deemed necessary. Generalized PODs '11 (415)

C++17 `pmr` allocator – a form of stateful memory allocator that supports allocating memory dynamically via the `std::pmr::memory_resource` *abstract base class*, thus providing a common vocabulary for enabling custom memory allocation, especially allocation from local arenas; see **lakos16**, **lakos17a**, **lakos17b**, **lakos19**. *Rvalue* References (763)

cache associativity – the characterization of the mapping from cache lines to locations in physical memory, ranging in implementation from one-to-many (direct mapped) to many-to-many with no restrictions (fully associative).

cache hit – successfully finding needed data resident in a given level of cache, obviating its retrieval from outside that cache. `alignas` (181)

cache line – the smallest unit of memory read from or written to main memory by a cache in a single operation. `alignas` (174), Generalized PODs '11 (459), `noexcept` Specifier (1142)

cache miss – failing to find needed data resident in a given level of cache, thus requiring its retrieval from outside that cache. `alignas` (182)

call operator – a nonstatic member function having the name `operator()` and taking zero or more arguments; see also callable object. Lambdas (574)

callable entity – one for which the call operator may be applied — e.g., function, callable object, reference to either, and pointer to function; see also invocable entity.