

Glossary

- template argument deduction** – the process by which template arguments are determined from the types of the *function* arguments when calling a function template. [Variadic Templates \(894\)](#)
- template argument list** – the sequence of arguments — types, templates, or values, depending on the corresponding parameters — that are used to specify explicit, nondeduced arguments to a template instantiation. [Variadic Templates \(882\)](#)
- template head** – the keyword **template** and associated template parameter list used to introduce the declaration or definition of a template. [Variable Templates \(157\)](#)
- template instantiation** – (1) the process of substituting template arguments into the template parameters of a template declaration or definition to produce a concrete entity declaration or definition, respectively, and (2) the entity produced by this process. [Forwarding References \(382\)](#)
- template-instantiation time** – the point at which, for a given template, the compiler performs template instantiation, triggered when encountering a point of instantiation for that template in the source code. Note that when a template definition is first encountered, certain semantic analysis and error detection — particularly those involving the template parameters — must be deferred until template-instantiation time. [static_assert \(116\)](#)
- template parameter** – one, for a given template, that is associated with a template argument (i.e., a type, template, or value) when that template is instantiated; see also point of instantiation. [Variadic Templates \(896\)](#), [friend '11 \(1031\)](#)
- template parameter list** – the list of template parameters, surrounded by angle brackets (< and >), in the template head of a template declaration or definition. [Variadic Templates \(888\)](#)
- template parameter pack** – a template parameter that accepts one or more template arguments, identified by an ellipsis (...) preceding the parameter name (if any) in the template head; the size of a parameter pack is determined by the number of template arguments supplied or deduced for the pack parameter at the point of instantiation for the template. [Generalized PODs '11 \(437\)](#), [Variadic Templates \(879\)](#)
- template template parameter** – a template parameter that expects a template argument that is itself a template: **template <template <typename> class X> class Y;** (declares a template template parameter, X, for a [template class, Y](#)). [Variable Templates \(165\)](#), [Variadic Templates \(902\)](#)
- template template parameter pack** – a template parameter pack of template template parameters: **template <template <typename> class X...>** (X is a template template parameter pack). [Variadic Templates \(903\)](#)
- temporary** – short for temporary object; see also expiring object. [initializer_list \(555\)](#), [Rvalue References \(724\)](#)
- temporary materialization** – the act, by the compiler, of creating a temporary object when needed, e.g., when binding a reference to a *prvalue*. [Rvalue References \(717\)](#)
- temporary object** – an unnamed object, created by evaluating an expression, whose lifetime generally extends until the end of the outermost enclosing expression in which the temporary is created. [Rvalue References \(711\)](#)
- ternary operator** – an operator, formed with ? and :, taking three operands: (1) a conditional expression before the ?, (2) an expression between the ? and the : to be evaluated to produce the result if the first argument is **true**, and (3) an expression after the : to be evaluated to produce the result if the first argument is **false**. The type of the complete expression is the common type of the second and third operands. [constexpr Functions \(268\)](#), [noexcept Operator \(615\)](#)