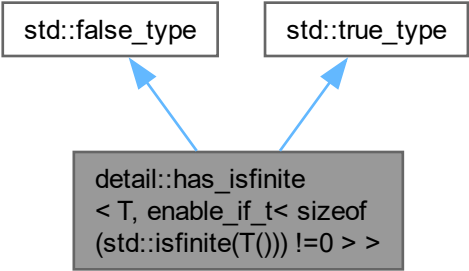


`std::false_type`

`std::true_type`

`detail::has_isfinite
< T, enable_if_t< sizeof
(std::isfinite(T())) !=0 > >`



```
graph BT; A["detail::has_isfinite< T, enable_if_t< sizeof (std::isfinite(T())) !=0 > >"] --> B["std::false_type"]; A --> C["std::true_type"];
```

The diagram illustrates the specialization of the `std::isfinite` trait. At the bottom, a gray box contains the definition of `detail::has_isfinite`, which is a template struct that inherits from `std::false_type` or `std::true_type` based on whether the type `T` has a non-zero `sizeof` for `std::isfinite(T())`. Two blue arrows point from this gray box to the `std::false_type` and `std::true_type` boxes above it, indicating that `detail::has_isfinite` inherits from both.