

`std::false_type`

`std::true_type`

`detail::is_back_insert
_iterator< basic_appender
< T > >`

```
graph BT; A["detail::is_back_insert<basic_appender<T>>"] --> B["std::false_type"]; A --> C["std::true_type"];
```

The diagram illustrates a C++ SFINAE (Substitution Failure Is Not An Error) scenario. A function template in the `detail` namespace, `is_back_insert_iterator`, is specialized for `basic_appender` with a template parameter `T`. This specialization is used to determine the value of `std::false_type` and `std::true_type`. The arrows indicate that the function template is a candidate for substitution into the `std::false_type` and `std::true_type` definitions.