

`std::false_type`

`std::conditional_t< bool(B1::value), B1, disjunction< Bn... > >`

`Catch::Detail::disjunction< B1, Bn... >`

```
graph BT; A[Catch::Detail::disjunction< B1, Bn... >] --> B[std::false_type]; A --> C[std::conditional_t< bool(B1::value), B1, disjunction< Bn... > >];
```

The diagram illustrates a relationship between three C++ type definitions. At the bottom is a box containing `Catch::Detail::disjunction< B1, Bn... >`. Two arrows originate from the top edge of this box. One arrow points to the bottom edge of the box containing `std::false_type` on the left. The other arrow points to the bottom edge of the box containing `std::conditional_t< bool(B1::value), B1, disjunction< Bn... > >` on the right. This suggests that `Catch::Detail::disjunction` is a base or specialization for both `std::false_type` and the `std::conditional_t` expression.