

```
frontend::common::BaseId  
Wrapper< size_t, IdWrapperLabel,  
label_, static_cast< size_t  
>(-1)>::InvalidId
```

```
frontend::common::Explicit  
IdWrapperCustomizeInvalidValue  
::InvalidId
```

```
frontend::common::NonExplicit  
IdWrapperCustomizeInvalidValue  
::InvalidId
```

```
frontend::common::BaseId  
Wrapper::InvalidValue
```

```
graph LR; A[frontend::common::BaseIdWrapper<...>::InvalidId] --> D[frontend::common::BaseIdWrapper::InvalidValue]; B[frontend::common::ExplicitIdWrapperCustomizeInvalidValue::InvalidId] --> D; C[frontend::common::NonExplicitIdWrapperCustomizeInvalidValue::InvalidId] --> D;
```

The diagram illustrates the relationship between different invalid value representations in the frontend. Three specialized or base classes on the left point to a common target on the right. The target is a static member variable of the base class. The three classes on the left are: 1. A template specialization of the base class's InvalidId. 2. An explicit wrapper's InvalidValue. 3. A non-explicit wrapper's InvalidValue.