

```
Eigen::internal::make  
_coherent_impl< A, Matrix  
< B_Scalar, B_Rows, B_Cols,  
B_Options, B_MaxRows, B_MaxCols  
> >::run
```

```
Eigen::internal::make  
_coherent_impl< Matrix  
< A_Scalar, A_Rows, A_Cols,  
A_Options, A_MaxRows, A_MaxCols  
>, B >::run
```

```
Eigen::internal::make  
_coherent_expression
```

```
Eigen::internal::Assignment  
< DstXprType, CwiseBinaryOp  
< internal::scalar_product  
_op< ScalarBis, Scalar >, const  
CwiseNullaryOp< internal::scalar  
_constant_op< ScalarBis >, Plain  
>, const Product< Lhs, Rhs, DefaultProduct  
> >, AssignFunc, Dense2Dense >::run
```

```
Eigen::internal::check  
_transpose_aliasing_run  
_time_selector< Scalar,  
_DestIsTransposed, CwiseBinaryOp  
< BinOp, DerivedA, DerivedB > >::run
```

```
Eigen::CwiseBinaryOp::lhs
```

```
graph LR; A["Eigen::internal::make_coherent_impl< A, Matrix< B_Scalar, B_Rows, B_Cols, B_Options, B_MaxRows, B_MaxCols > >::run"] --> C["Eigen::internal::make_coherent_expression"]; B["Eigen::internal::make_coherent_impl< Matrix< A_Scalar, A_Rows, A_Cols, A_Options, A_MaxRows, A_MaxCols >, B >::run"] --> C; C --> D["Eigen::internal::Assignment< DstXprType, CwiseBinaryOp< internal::scalar_product_op< ScalarBis, Scalar >, const CwiseNullaryOp< internal::scalar_constant_op< ScalarBis >, Plain >, const Product< Lhs, Rhs, DefaultProduct > >, AssignFunc, Dense2Dense >::run"]; D --> E["Eigen::CwiseBinaryOp::lhs"]; F["Eigen::internal::check_transpose_aliasing_run_time_selector< Scalar, _DestIsTransposed, CwiseBinaryOp< BinOp, DerivedA, DerivedB > >::run"] --> E; style E fill:#808080
```