RInside Version 0.2.16

Generated by Doxygen 1.8.13

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 binary_stream_helpers Namespace Reference

Classes

- struct has_serialization_members
- struct has_serialization_members_cv
- struct has_serialization_members_cv< T, void_t< serialize_member_t< T >, deserialize_member_t< T > > >
- struct void_t_struct

Typedefs

- template<typename... C>
  using void_t = typename void_t_struct< C... >::type
- template<typename T >
  using serialize_member_t = decltype(std::declval< T & >().serialize(std::declval< BinaryStream & >()))
- template<typename T >
  using deserialize_member_t = decltype(T::deserialize(std::declval< BinaryStream & >()))

Functions

- template<typename T >
  std::enable_if< std::is_arithmetic< T >::value >::type stream_write (BinaryStream &stream, T &t)
- template<typename T >
  std::enable_if< has_typeid< T >::value &&std::is_class< T >::value &&has_serialization_members< T >::value >::type stream_write (BinaryStream &stream, T &t)
- template<typename T >
  std::enable_if< has_typeid< T >::value &&std::is_class< T >::value &&!has_serialization_members< T >↩
  ::value >::type stream_write (BinaryStream &stream, T &t)
- template<typename T >
  std::enable_if< std::is_arithmetic< T >::value, T >::type stream_read (BinaryStream &stream)
- template<typename T >
  std::enable_if< has_typeid< T >::value &&std::is_class< T >::value &&has_serialization_members< T >↩
  ::value, T >::type stream_read (BinaryStream &stream)
- template<typename T >
  std::enable_if< has_typeid< T >::value &&std::is_class< T >::value &&!has_serialization_members< T >↩
  ::value, T >::type stream_read (BinaryStream &stream)

5.1.1 Typedef Documentation

5.1.1.1 deserialize_member_t

```
template<typename T >
using binary_stream_helpers::deserialize_member_t = typedef decltype( T::deserialize( std::declval<Binary←
Stream&>() ) )
```

Definition at line 116 of file binarystream.h.

5.1.1.2 serialize_member_t

```
template<typename T >
using binary_stream_helpers::serialize_member_t = typedef decltype( std::declval<T&>().serialize(
std::declval<BinaryStream&>() ) )
```

Definition at line 113 of file binarystream.h.

5.1.1.3 void_t

```
template<typename... C>
using binary_stream_helpers::void_t = typedef typename void_t_struct<C...>::type
```

Definition at line 107 of file binarystream.h.

5.1.2 Function Documentation

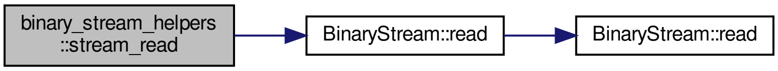
5.1.2.1 stream_read() [1/3]

```
template<typename T >
std::enable_if< std::is_arithmetic<T>::value, T >::type binary_stream_helpers::stream_read (
             BinaryStream & stream )
```

Definition at line 155 of file binarystream.h.

References BinaryStream::read().

Here is the call graph for this function:



5.1.2.2 stream_read() [2/3]

```
template<typename T >
std::enable_if< has_typeid<T>::value && std::is_class<T>::value && has_serialization_members<T>↩
::value, T >::type binary_stream_helpers::stream_read (
             BinaryStream & stream )
```

Definition at line 163 of file binarystream.h.

5.1.2.3 stream_read() [3/3]

```
template<typename T >
std::enable_if< has_typeid<T>::value && std::is_class<T>::value && !has_serialization_members<T>↩
::value, T >::type binary_stream_helpers::stream_read (
             BinaryStream & stream )
```

Definition at line 169 of file binarystream.h.

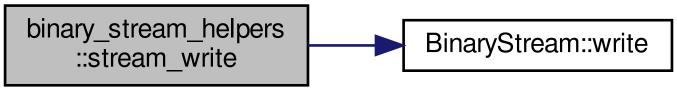
5.1.2.4 stream_write() [1/3]

```
template<typename T >
std::enable_if< std::is_arithmetic<T>::value >::type binary_stream_helpers::stream_write (
            BinaryStream & stream,
            T & t )
```

Definition at line 133 of file binarystream.h.

References BinaryStream::write().

Here is the call graph for this function:



5.1.2.5 stream_write() [2/3]

```
template<typename T >
std::enable_if< has_typeid<T>::value && std::is_class<T>::value && has_serialization_members<T>↩
::value >::type binary_stream_helpers::stream_write (
            BinaryStream & stream,
            T & t )
```

Definition at line 139 of file binarystream.h.

5.1.2.6 stream_write() [3/3]

```
template<typename T >
std::enable_if< has_typeid<T>::value && std::is_class<T>::value && !has_serialization_members<T>↩
::value >::type binary_stream_helpers::stream_write (
            BinaryStream & stream,
            T & t )
```

Definition at line 145 of file binarystream.h.

5.2 callback_helper Namespace Reference

Classes

- class parameter_error_exception
- struct send_pack
- struct send_pack< First, Remaining... >
- struct send_pack<>
- class type_mismatch_exception

Functions

- template<typename T >
  T read_from_stream_with_typeid (BinaryStream &stream)
- template<typename RESULT_TYPE >
  void call (const std::function< RESULT_TYPE()> &fun, BinaryStream &stream)
- template<typename RESULT_TYPE , typename U0 >
  void call (const std::function< RESULT_TYPE(U0)> &fun, BinaryStream &stream)
- template<typename RESULT_TYPE , typename U0 , typename U1 >
  void call (const std::function< RESULT_TYPE(U0, U1)> &fun, BinaryStream &stream)
- template<typename RESULT_TYPE , typename U0 , typename U1 , typename U2 >
  void call (const std::function< RESULT_TYPE(U0, U1, U2)> &fun, BinaryStream &stream)

5.2.1 Function Documentation

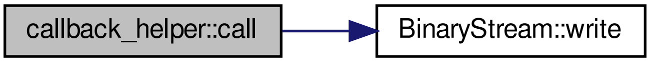
5.2.1.1 call() [1/4]

```
template<typename RESULT_TYPE >
void callback_helper::call (
            const std::function< RESULT_TYPE()> & fun,
            BinaryStream & stream )
```

Definition at line 57 of file callback_helper.h.

References BinaryStream::write().

Here is the call graph for this function:



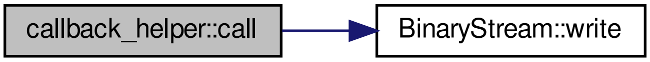
5.2.1.2 call() [2/4]

```
template<typename RESULT_TYPE , typename U0 >
void callback_helper::call (
             const std::function< RESULT_TYPE(U0)> & fun,
             BinaryStream & stream )
```

Definition at line 64 of file callback_helper.h.

References BinaryStream::write().

Here is the call graph for this function:



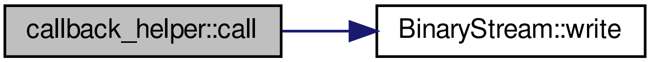
5.2.1.3 call() [3/4]

```
template<typename RESULT_TYPE , typename U0 , typename U1 >
void callback_helper::call (
             const std::function< RESULT_TYPE(U0, U1)> & fun,
             BinaryStream & stream )
```

Definition at line 73 of file callback_helper.h.

References BinaryStream::write().

Here is the call graph for this function:



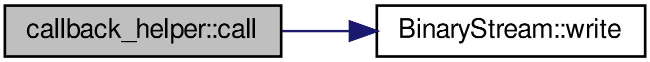
5.2.1.4 call() [4/4]

```
template<typename RESULT_TYPE , typename U0 , typename U1 , typename U2 >
void callback_helper::call (
             const std::function< RESULT_TYPE(U0, U1, U2)> & fun,
             BinaryStream & stream )
```

Definition at line 83 of file callback_helper.h.

References BinaryStream::write().

Here is the call graph for this function:



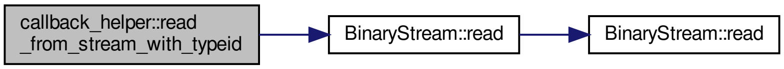
5.2.1.5 read_from_stream_with_typeid()

```
template<typename T >
T callback_helper::read_from_stream_with_typeid (
             BinaryStream & stream )
```

Definition at line 38 of file callback_helper.h.

References BinaryStream::read(), RIS_REPLY_ERROR, and RIS_REPLY_VALUE.

Here is the call graph for this function:



5.3 Rcpp Namespace Reference

Typedefs

- typedef InternalFunctionForRInsideServer_Impl< PreserveStorage > InternalFunctionForRInsideServer

Functions

- template<>
  SEXP wrap (const Bar &bar)
- template<>
  Bar as (SEXP sexp)
- template<>
  SEXP wrap (const Foo &foo)
- template<>
  Foo as (SEXP sexp)
- RCPP_API_CLASS (InternalFunctionForRInsideServer_Impl)

5.3.1 Typedef Documentation

5.3.1.1 InternalFunctionForRInsideServer

```
typedef InternalFunctionForRInsideServer_Impl<PreserveStorage> Rcpp::InternalFunctionForRInside↩
Server
```

Definition at line 78 of file internalfunction_clone.h.

5.3.2 Function Documentation

5.3.2.1 as() [1/2]

```
template<>
Foo Rcpp::as (
                SEXP sexp )
```

Definition at line 19 of file foo_rcpp_wrapper_definitions.h.

5.3.2.2 as() [2/2]

```
template<>
Bar Rcpp::as (
                SEXP sexp )
```

Definition at line 19 of file bar_rcpp_wrapper_definitions.h.

Referenced by main().

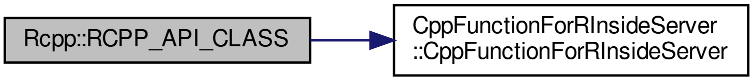
5.3.2.3 RCPP_API_CLASS()

```
Rcpp::RCPP_API_CLASS (
             InternalFunctionForRInsideServer_Impl  )
```

Definition at line 60 of file internalfunction_clone.h.

References CppFunctionForRInsideServer::callback_id, CppFunctionForRInsideServer::CppFunctionForRInsideServer(), CppFunctionForRInsideServer::server, and CppFunctionForRInsideServer::types.

Here is the call graph for this function:



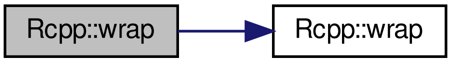
5.3.2.4 wrap() [1/2]

```
template<>
SEXP Rcpp::wrap (
             const Foo & foo )
```

Definition at line 10 of file foo_rcpp_wrapper_definitions.h.

References Foo::a, Foo::b, Foo::name, and wrap().

Here is the call graph for this function:



5.3.2.5 wrap() [2/2]

```
template<>
SEXP Rcpp::wrap (
             const Bar & bar )
```

Definition at line 11 of file bar_rcpp_wrapper_definitions.h.

References Bar::foo, and Bar::name.

Referenced by main(), and wrap().

5.4 serialization Namespace Reference

Classes

- struct serializer
- struct serializer< std::string >
- struct serializer< std::vector< T > >

5.5 typeid_helpers Namespace Reference

Classes

- struct has_typeid_member
- struct has_typeid_member< T, void_t< decltype(T::TYPEID)> >
- struct id
- struct id< double, void >
- struct id< float, void >
- struct id< int16_t, void >
- struct id< int32_t, void >
- struct id< int64_t, void >
- struct id< int8_t, void >
- struct id< std::string, void >
- struct id< std::vector< double >, void >
- struct id< std::vector< float >, void >
- struct id< std::vector< int16_t >, void >
- struct id< std::vector< int32_t >, void >
- struct id< std::vector< int64_t >, void >
- struct id< std::vector< int8_t >, void >
- struct id< std::vector< std::string >, void >
- struct id< std::vector< uint16_t >, void >
- struct id< std::vector< uint32_t >, void >
- struct id< std::vector< uint64_t >, void >
- struct id< std::vector< uint8_t >, void >
- struct id< T, typename std::enable_if< has_typeid_member< T >::value >::type >
- struct id< uint16_t, void >
- struct id< uint32_t, void >
- struct id< uint64_t, void >
- struct id< uint8_t, void >
- struct id< void, void >
- struct void_t_struct

Typedefs

- template<typename... C>
  using void_t = typename void_t_struct< C... >::type

5.5.1 Typedef Documentation

5.5.1.1 void_t

```
template<typename...  C>
using typeid_helpers::void_t = typedef typename void_t_struct<C...>::type
```

Definition at line 28 of file typeid.h.

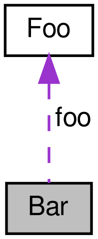
Chapter 6

Class Documentation

6.1 Bar Class Reference

```
#include <bar.h>
```

Collaboration diagram for Bar:



Public Member Functions

- Bar (const std::string &name, const Foo &foo)
- ∼Bar ()
- void serialize (BinaryStream &stream) const

Static Public Member Functions

- static Bar deserialize (BinaryStream &stream)

Public Attributes

- std::string name
- Foo foo

Static Public Attributes

- static const int32_t TYPEID = 2

6.1.1 Detailed Description

Definition at line 19 of file bar.h.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 Bar()

```
Bar::Bar (
            const std::string & name,
            const Foo & foo )
```

Definition at line 3 of file bar.cpp.

Referenced by deserialize().

6.1.2.2 ∼Bar()

```
Bar::∼Bar ( )
```

Definition at line 6 of file bar.cpp.

6.1.3 Member Function Documentation

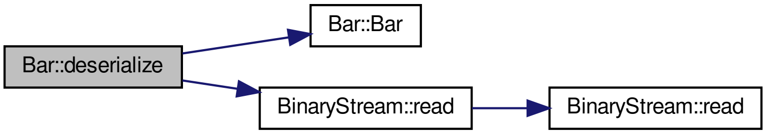
6.1.3.1 deserialize()

```
Bar Bar::deserialize (
             BinaryStream & stream )  [static]
```

Definition at line 15 of file bar.cpp.

References Bar(), foo, name, and BinaryStream::read().

Here is the call graph for this function:



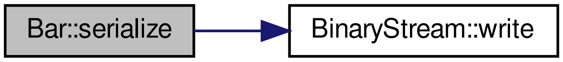
6.1.3.2 serialize()

```
void Bar::serialize (
             BinaryStream & stream ) const
```

Definition at line 10 of file bar.cpp.

References foo, name, and BinaryStream::write().

Here is the call graph for this function:



6.1.4 Member Data Documentation

6.1.4.1 foo

```
Foo Bar::foo
```

Definition at line 25 of file bar.h.

Referenced by deserialize(), serialize(), and Rcpp::wrap().

6.1.4.2 name

```
std::string Bar::name
```

Definition at line 24 of file bar.h.

Referenced by deserialize(), serialize(), and Rcpp::wrap().

6.1.4.3 TYPEID

```
const int32_t Bar::TYPEID = 2  [static]
```

Definition at line 28 of file bar.h.

The documentation for this class was generated from the following files:

- inst/examples/sandboxed_server/datatypes/bar.h
- inst/examples/sandboxed_server/datatypes/bar.cpp

6.2 BinaryStream Class Reference

```
#include <binarystream.h>
```

Classes

- class stream_exception

Public Member Functions

- BinaryStream (int read_fd, int write_fd)
- ∼BinaryStream ()
- void close ()
- BinaryStream (const BinaryStream &)=delete
- BinaryStream & operator= (const BinaryStream &)=delete
- BinaryStream (BinaryStream &&)
- BinaryStream & operator= (BinaryStream &&)
- void write (const char ∗buffer, size_t len)
- template< typename T >
  void write (const T &t)
- template< typename T >
  void write (T &t)
- size_t read (char ∗buffer, size_t len)
- template< typename T >
  std::enable_if< std::is_arithmetic< T >::value, size_t >::type read (T ∗t)
- template< typename T >
  T read ()

Static Public Member Functions

- static BinaryStream connectToUnixSocket (const char ∗)

Private Attributes

- bool is_eof
- int read_fd
- int write_fd

6.2.1 Detailed Description

Definition at line 45 of file binarystream.h.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 BinaryStream() [1/3]

```
BinaryStream::BinaryStream (
             int read_fd,
             int write_fd )
```

Definition at line 18 of file binarystream.cpp.

Referenced by connectToUnixSocket().

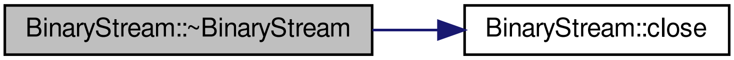
6.2.2.2 ∼BinaryStream()

```
BinaryStream::∼BinaryStream ( )
```

Definition at line 21 of file binarystream.cpp.

References close().

Here is the call graph for this function:



6.2.2.3 BinaryStream() [2/3]

```
BinaryStream::BinaryStream (
             const BinaryStream &  )  [delete]
```

6.2.2.4 BinaryStream() [3/3]

```
BinaryStream::BinaryStream (
             BinaryStream && other )
```

Definition at line 39 of file binarystream.cpp.

6.2.3 Member Function Documentation

6.2.3.1 close()

```
void BinaryStream::close ( )
```

Definition at line 25 of file binarystream.cpp.

References is_eof, read_fd, and write_fd.

Referenced by connectToUnixSocket(), and ∼BinaryStream().

6.2.3.2 connectToUnixSocket()

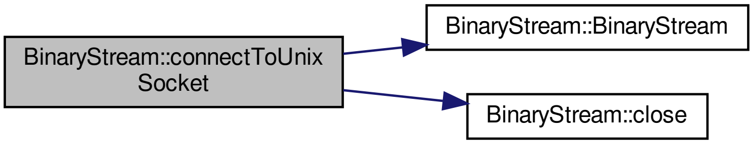
```
BinaryStream BinaryStream::connectToUnixSocket (
             const char * server_path )  [static]
```

Definition at line 53 of file binarystream.cpp.

References BinaryStream(), and close().

Referenced by test_callbacks(), test_console_output(), test_multiple(), test_plot(), and test_setting_getting().

Here is the call graph for this function:


BinaryStream::connectToUnix Socket
BinaryStream::BinaryStream
BinaryStream::close

6.2.3.3 operator=() [1/2]

```
BinaryStream& BinaryStream::operator= (
             const BinaryStream &  )  [delete]
```

6.2.3.4 operator=() [2/2]

```
BinaryStream & BinaryStream::operator= (
             BinaryStream && other )
```

Definition at line 45 of file binarystream.cpp.

References is_eof, read_fd, and write_fd.

6.2.3.5 read() [1/3]

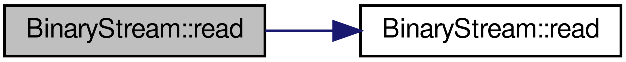
```
size_t BinaryStream::read (
              char * buffer,
              size_t len )
```

Definition at line 84 of file binarystream.cpp.

References is_eof, read(), and read_fd.

Referenced by Foo::deserialize(), Bar::deserialize(), serialization::serializer< std::string >::deserialize(), serialization::serializer< std::vector< T > >::deserialize(), RInsideClient::getConsoleOutput(), RInsideClient::getPlot(), RInsideClient::getValue(), RInsideClient::parseEval(), callback_helper::read_from_stream_with_typeid(), RInsideClient::readReply(), RInsideServer::registerType(), RInsideServer::run(), RInsideClient::runScript(), RInsideServer::sexp_from_stream(), and binary_stream_helpers::stream_read().

Here is the call graph for this function:



6.2.3.6 read() [2/3]

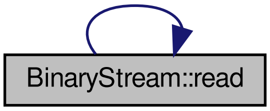
```
template<typename T >
std::enable_if< std::is_arithmetic<T>::value, size_t>::type BinaryStream::read (
              T * t )  [inline]
```

Definition at line 64 of file binarystream.h.

References read().

Referenced by read().

Here is the call graph for this function:



6.2.3.7 read() [3/3]

```
template<typename T >
T BinaryStream::read ( )
```

Definition at line 182 of file binarystream.h.

Referenced by read().

6.2.3.8 write() [1/3]

```
void BinaryStream::write (
             const char * buffer,
             size_t len )
```

Definition at line 72 of file binarystream.cpp.

References write_fd.

Referenced by callback_helper::call(), RInsideClient::getValue(), RInsideClient::initPlot(), CppFunctionForRInsideServer::operator()(), RInsideServer::registerType(), RInsideClient::RInsideClient(), RInsideServer::run(), RInsideClient::runScript(), callback_helper::send_pack< First, Remaining... >::send(), RInsideServer::sendReply(), Foo::serialize(), Bar::serialize(), serialization::serializer< std::string >::serialize(), serialization::serializer< std::vector< T > >::serialize(), RInsideClient::setCallback(), RInsideClient::setValue(), binary_stream_helpers::stream_write(), RInsideClient::writeCommand(), and RInsideClient::∼RInsideClient().

6.2.3.9 write() [2/3]

```
template<typename T >
void BinaryStream::write (
             const T & t )
```

Definition at line 175 of file binarystream.h.

6.2.3.10 write() [3/3]

```
template<typename T >
void BinaryStream::write (
             T & t )
```

Definition at line 178 of file binarystream.h.

6.2.4 Member Data Documentation

6.2.4.1 is_eof

```
bool BinaryStream::is_eof  [private]
```

Definition at line 72 of file binarystream.h.

Referenced by close(), operator=(), and read().

6.2.4.2 read_fd

```
int BinaryStream::read_fd  [private]
```

Definition at line 73 of file binarystream.h.

Referenced by close(), operator=(), and read().

6.2.4.3 write_fd

```
int BinaryStream::write_fd  [private]
```

Definition at line 73 of file binarystream.h.

Referenced by close(), operator=(), and write().

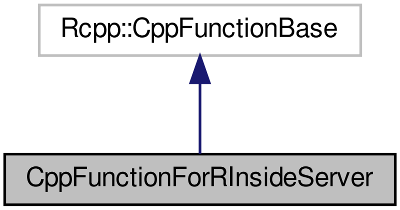
The documentation for this class was generated from the following files:

- inst/examples/sandboxed_server/common/binarystream.h
- inst/examples/sandboxed_server/common/binarystream.cpp

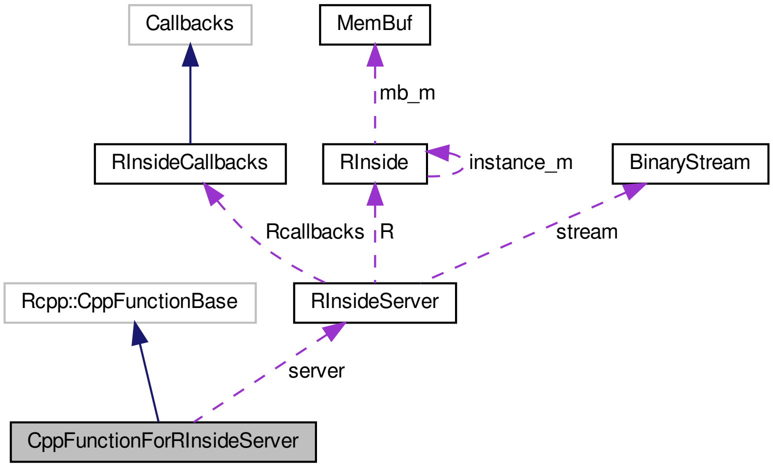
6.3 CppFunctionForRInsideServer Class Reference

```
#include <internalfunction_clone.h>
```

Inheritance diagram for CppFunctionForRInsideServer:



Collaboration diagram for CppFunctionForRInsideServer:



Public Member Functions

- CppFunctionForRInsideServer (RInsideServer &server, uint32_t callback_id, const std::vector< int32_t > &types)
- virtual ~CppFunctionForRInsideServer ()
- SEXP operator() (SEXP ∗args)

Private Attributes

- RInsideServer & server
- uint32_t callback_id
- const std::vector< int32_t > types

6.3.1 Detailed Description

Definition at line 13 of file internalfunction_clone.h.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 CppFunctionForRInsideServer()

```
CppFunctionForRInsideServer::CppFunctionForRInsideServer (
            RInsideServer & server,
            uint32_t callback_id,
            const std::vector< int32_t > & types )  [inline]
```

Definition at line 15 of file internalfunction_clone.h.

Referenced by Rcpp::RCPP_API_CLASS().

6.3.2.2 ∼CppFunctionForRInsideServer()

```
virtual CppFunctionForRInsideServer::∼CppFunctionForRInsideServer ( )  [inline], [virtual]
```

Definition at line 17 of file internalfunction_clone.h.

6.3.3 Member Function Documentation

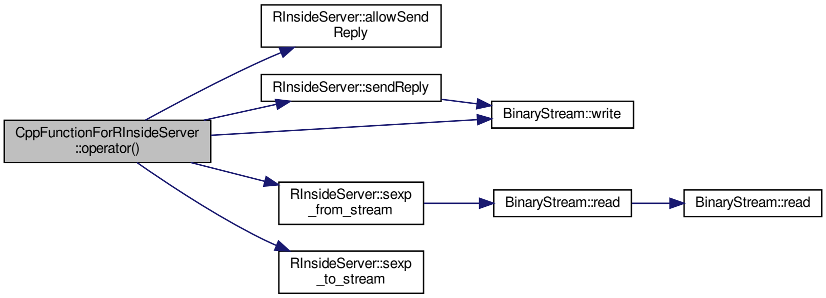
6.3.3.1 operator()()

```
SEXP CppFunctionForRInsideServer::operator() (
             SEXP * args )  [inline]
```

Definition at line 19 of file internalfunction_clone.h.

References RInsideServer::allowSendReply(), callback_id, LOG, RIS_REPLY_CALLBACK, RInsideServer::send↩Reply(), server, RInsideServer::sexp_from_stream(), RInsideServer::sexp_to_stream(), RInsideServer::stream, types, and BinaryStream::write().

Here is the call graph for this function:


RInsideServer::allowSend Reply
RInsideServer::sendReply
BinaryStream::write
CppFunctionForRInsideServer ::operator()
RInsideServer::sexp _from_stream
BinaryStream::read
BinaryStream::read
RInsideServer::sexp _to_stream

6.3.4 Member Data Documentation

6.3.4.1 callback_id

```
uint32_t CppFunctionForRInsideServer::callback_id  [private]
```

Definition at line 49 of file internalfunction_clone.h.

Referenced by operator()(), and Rcpp::RCPP_API_CLASS().

6.3.4.2 server

```
RInsideServer& CppFunctionForRInsideServer::server  [private]
```

Definition at line 48 of file internalfunction_clone.h.

Referenced by operator()(), and Rcpp::RCPP_API_CLASS().

6.3.4.3 types

```
const std::vector<int32_t> CppFunctionForRInsideServer::types  [private]
```

Definition at line 50 of file internalfunction_clone.h.

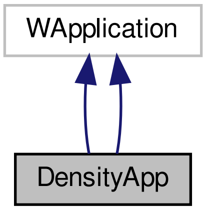
Referenced by operator()(), and Rcpp::RCPP_API_CLASS().

The documentation for this class was generated from the following file:

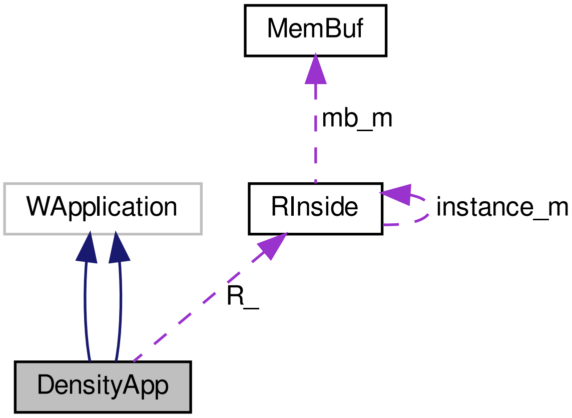
- inst/examples/sandboxed_server/server/internalfunction_clone.h

6.4 DensityApp Class Reference

Inheritance diagram for DensityApp:



Collaboration diagram for DensityApp:



Public Member Functions

- DensityApp (const WEnvironment &env, RInside &R)
- DensityApp (const WEnvironment &env, RInside &R)

Private Types

- enum Kernel {
  Gaussian = 0, Epanechnikov = 1, Rectangular = 2, Triangular = 3,
  Cosine = 4, Gaussian = 0, Epanechnikov = 1, Rectangular = 2,
  Triangular = 3, Cosine = 4 }
- enum Kernel {
  Gaussian = 0, Epanechnikov = 1, Rectangular = 2, Triangular = 3,
  Cosine = 4, Gaussian = 0, Epanechnikov = 1, Rectangular = 2,
  Triangular = 3, Cosine = 4 }

Private Member Functions

- void reportButton ()
- void reportEdit ()
- void reportSpinner ()
- void plot ()
- void reportButton ()
- void reportEdit ()
- void reportSpinner ()
- void plot ()

Private Attributes

- WLineEdit ∗ codeEdit_
- WButtonGroup ∗ group_
- WSpinBox ∗ spin_
- WImage ∗ img_
- WFileResource ∗ imgfile_
- WText ∗ greeting_
- RInside & R_
- std::string tempfile_
- int bw_
- int kernel_
- std::string cmd_
- Rcpp::NumericVector Yvec_

6.4.1 Detailed Description

Definition at line 33 of file wtdensity.cpp.

6.4.2 Member Enumeration Documentation

6.4.2.1 Kernel [1/2]

```
enum DensityApp::Kernel  [private]
```

Enumerator

Gaussian	
Epanechnikov	
Rectangular	
Triangular	
Cosine	
Gaussian	
Epanechnikov	
Rectangular	
Triangular	
Cosine	

Definition at line 50 of file wtdensity.cpp.

6.4.2.2 Kernel [2/2]

```
enum DensityApp::Kernel  [private]
```

Enumerator

Gaussian	
Epanechnikov	
Rectangular	
Triangular	
Cosine	
Gaussian	
Epanechnikov	
Rectangular	
Triangular	
Cosine	

Definition at line 50 of file wtdensityPlain.cpp.

6.4.3 Constructor & Destructor Documentation

6.4.3.1 DensityApp() [1/2]

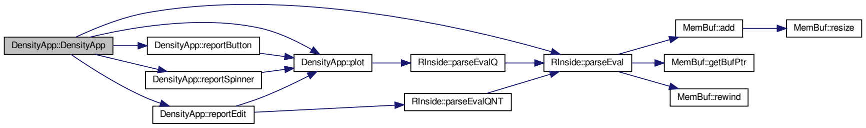
```
DensityApp::DensityApp (
             const WEnvironment & env,
             RInside & R )
```

Definition at line 63 of file wtdensity.cpp.

References bw_, cmd_, codeEdit_, Cosine, Epanechnikov, Gaussian, greeting_, group_, img_, imgfile_, kernel_, R↩ Inside::parseEval(), plot(), R_, Rectangular, reportButton(), reportEdit(), reportSpinner(), spin_, tempfile_, and Triangular.

Referenced by createApplication().

Here is the call graph for this function:



6.4.3.2 DensityApp() [2/2]

```
DensityApp::DensityApp (
             const WEnvironment & env,
             RInside & R )
```

6.4.4 Member Function Documentation

6.4.4.1 plot() [1/2]

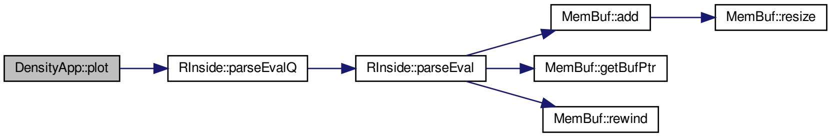
```
void DensityApp::plot (
             void ) [private]
```

Definition at line 169 of file wtdensity.cpp.

References bw_, greeting_, imgfile_, kernel_, RInside::parseEvalQ(), R_, tempfile_, and Yvec_.

Referenced by DensityApp(), reportButton(), reportEdit(), and reportSpinner().

Here is the call graph for this function:



6.4.4.2 plot() [2/2]

```
void DensityApp::plot ( )  [private]
```

6.4.4.3 reportButton() [1/2]

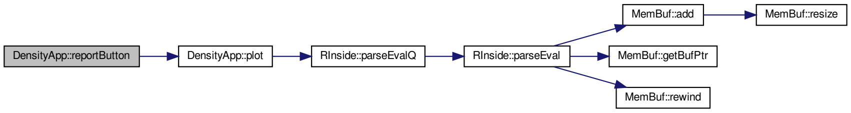
```
void DensityApp::reportButton ( )  [private]
```

Definition at line 151 of file wtdensity.cpp.

References group_, kernel_, and plot().

Referenced by DensityApp().

Here is the call graph for this function:



6.4.4.4 reportButton() [2/2]

```
void DensityApp::reportButton ( )  [private]
```

6.4.4.5 reportEdit() [1/2]

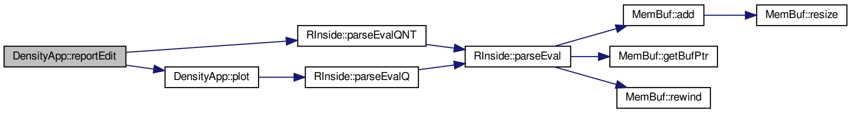
```
void DensityApp::reportEdit ( )  [private]
```

Definition at line 156 of file wtdensity.cpp.

References cmd_, codeEdit_, RInside::parseEvalQNT(), plot(), R_, and Yvec_.

Referenced by DensityApp().

Here is the call graph for this function:



6.4.4.6 reportEdit() [2/2]

```
void DensityApp::reportEdit ( )  [private]
```

6.4.4.7 reportSpinner() [1/2]

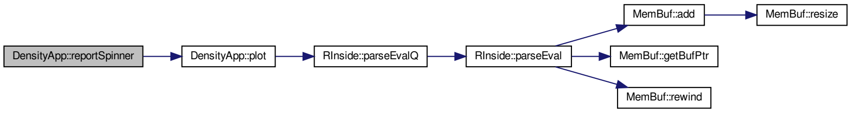
```
void DensityApp::reportSpinner ( )  [private]
```

Definition at line 164 of file wtdensity.cpp.

References bw_, plot(), and spin_.

Referenced by DensityApp().

Here is the call graph for this function:



6.4.4.8 reportSpinner() [2/2]

```
void DensityApp::reportSpinner ( )  [private]
```

6.4.5 Member Data Documentation

6.4.5.1 bw_

```
int DensityApp::bw_  [private]
```

Definition at line 55 of file wtdensity.cpp.

Referenced by DensityApp(), plot(), and reportSpinner().

6.4.5.2 cmd_

```
std::string DensityApp::cmd_  [private]
```

Definition at line 56 of file wtdensity.cpp.

Referenced by DensityApp(), and reportEdit().

6.4.5.3 codeEdit_

```
WLineEdit * DensityApp::codeEdit_  [private]
```

Definition at line 38 of file wtdensity.cpp.

Referenced by DensityApp(), and reportEdit().

6.4.5.4 greeting_

```
WText * DensityApp::greeting_  [private]
```

Definition at line 43 of file wtdensity.cpp.

Referenced by DensityApp(), and plot().

6.4.5.5 group_

```
WButtonGroup * DensityApp::group_  [private]
```

Definition at line 39 of file wtdensity.cpp.

Referenced by DensityApp(), and reportButton().

6.4.5.6 img_

```
WImage * DensityApp::img_  [private]
```

Definition at line 41 of file wtdensity.cpp.

Referenced by DensityApp().

6.4.5.7 imgfile_

```
WFileResource * DensityApp::imgfile_  [private]
```

Definition at line 42 of file wtdensity.cpp.

Referenced by DensityApp(), and plot().

6.4.5.8 kernel_

```
int DensityApp::kernel_  [private]
```

Definition at line 55 of file wtdensity.cpp.

Referenced by DensityApp(), plot(), and reportButton().

6.4.5.9 R_

```
RInside & DensityApp::R_  [private]
```

Definition at line 53 of file wtdensity.cpp.

Referenced by DensityApp(), plot(), and reportEdit().

6.4.5.10 spin_

```
WSpinBox * DensityApp::spin_  [private]
```

Definition at line 40 of file wtdensity.cpp.

Referenced by DensityApp(), and reportSpinner().

6.4.5.11 tempfile_

```
std::string DensityApp::tempfile_  [private]
```

Definition at line 54 of file wtdensity.cpp.

Referenced by DensityApp(), and plot().

6.4.5.12 Yvec_

```
Rcpp::NumericVector DensityApp::Yvec_  [private]
```

Definition at line 57 of file wtdensity.cpp.

Referenced by plot(), and reportEdit().

The documentation for this class was generated from the following files:

- inst/examples/wt/wtdensity.cpp
- inst/examples/wt/wtdensityPlain.cpp

6.5 Foo Class Reference

```
#include <foo.h>
```

Public Member Functions

- Foo (const std::string &name, int32_t a, int32_t b)
- ∼Foo ()
- void serialize (BinaryStream &stream) const
- Foo (int a, int b)
- ∼Foo ()

Static Public Member Functions

- static Foo deserialize (BinaryStream &stream)

Public Attributes

- std::string name
- int32_t a
- int32_t b
- int a
- int b

Static Public Attributes

- static const int32_t TYPEID = 1

6.5.1 Detailed Description

Definition at line 17 of file foo.h.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 Foo() [1/2]

```
Foo::Foo (
            const std::string & name,
            int32_t a,
            int32_t b )
```

Definition at line 4 of file foo.cpp.

Referenced by deserialize(), and main().

6.5.2.2 ∼Foo() [1/2]

```
Foo::∼Foo ( )
```

Definition at line 8 of file foo.cpp.

Referenced by main().

6.5.2.3 Foo() [2/2]

```
Foo::Foo (
            int a,
            int b )  [inline]
```

Definition at line 15 of file rinside_sample16.cpp.

6.5.2.4 ∼Foo() [2/2]

```
Foo::∼Foo ( )  [inline]
```

Definition at line 17 of file rinside_sample16.cpp.

6.5.3 Member Function Documentation

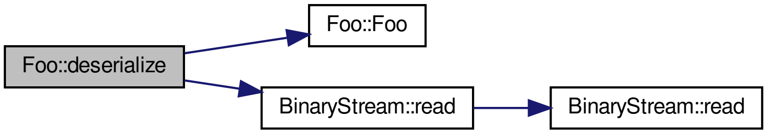
6.5.3.1 deserialize()

```
Foo Foo::deserialize (
             BinaryStream & stream )  [static]
```

Definition at line 19 of file foo.cpp.

References a, b, Foo(), name, and BinaryStream::read().

Here is the call graph for this function:



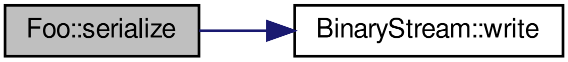
6.5.3.2 serialize()

```
void Foo::serialize (
             BinaryStream & stream ) const
```

Definition at line 12 of file foo.cpp.

References a, b, name, and BinaryStream::write().

Here is the call graph for this function:



6.5.4 Member Data Documentation

6.5.4.1 a [1/2]

```
int Foo::a
```

Definition at line 22 of file rinside_sample16.cpp.

6.5.4.2 a [2/2]

```
int32_t Foo::a
```

Definition at line 23 of file foo.h.

Referenced by deserialize(), main(), serialize(), swapFoo(), test_setting_getting(), and Rcpp::wrap().

6.5.4.3 b [1/2]

```
int Foo::b
```

Definition at line 22 of file rinside_sample16.cpp.

6.5.4.4 b [2/2]

```
int32_t Foo::b
```

Definition at line 23 of file foo.h.

Referenced by deserialize(), main(), serialize(), swapFoo(), test_setting_getting(), and Rcpp::wrap().

6.5.4.5 name

```
std::string Foo::name
```

Definition at line 22 of file foo.h.

Referenced by deserialize(), serialize(), test_setting_getting(), and Rcpp::wrap().

6.5.4.6 TYPEID

```
const int32_t Foo::TYPEID = 1  [static]
```

Definition at line 26 of file foo.h.

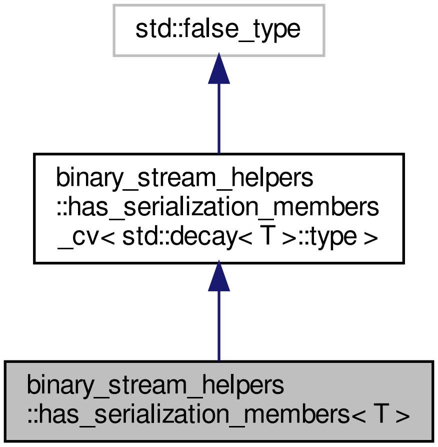
The documentation for this class was generated from the following files:

- inst/examples/sandboxed_server/datatypes/foo.h
- inst/examples/standard/rinside_sample16.cpp
- inst/examples/sandboxed_server/datatypes/foo.cpp

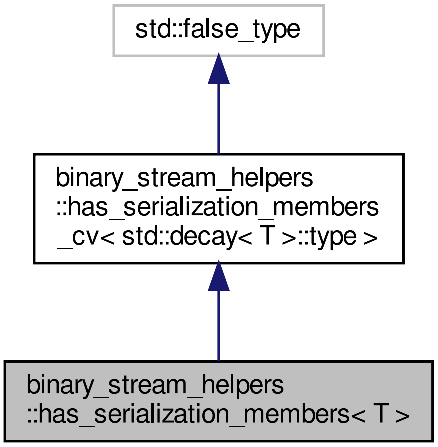
6.6 binary_stream_helpers::has_serialization_members< T > Struct Template Reference

```
#include <binarystream.h>
```

Inheritance diagram for binary_stream_helpers::has_serialization_members< T >:


std::false_type
binary_stream_helpers ::has_serialization_members _cv< std::decay< T >::type >
binary_stream_helpers ::has_serialization_members< T >

Collaboration diagram for binary_stream_helpers::has_serialization_members< T >:


std::false_type
binary_stream_helpers ::has_serialization_members _cv< std::decay< T >::type >
binary_stream_helpers ::has_serialization_members< T >

6.6.1 Detailed Description

template<typename T>
struct binary_stream_helpers::has_serialization_members< T >

Definition at line 126 of file binarystream.h.

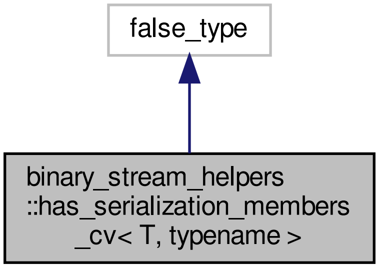
The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/binarystream.h

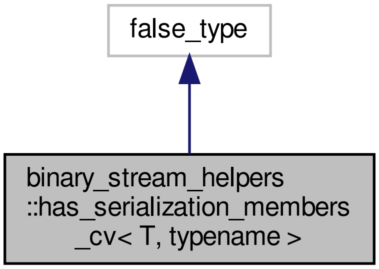
6.7 binary_stream_helpers::has_serialization_members_cv< T, typename > Struct Template Reference

```
#include <binarystream.h>
```

Inheritance diagram for binary_stream_helpers::has_serialization_members_cv< T, typename >:


false_type
binary_stream_helpers ::has_serialization_members _cv< T, typename >

Collaboration diagram for binary_stream_helpers::has_serialization_members_cv< T, typename >:


false_type
binary_stream_helpers ::has_serialization_members _cv< T, typename >

6.7.1 Detailed Description

template<typename T, typename = void>
struct binary_stream_helpers::has_serialization_members_cv< T, typename >

Definition at line 119 of file binarystream.h.

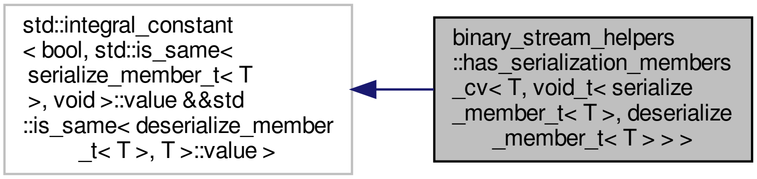
The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/binarystream.h

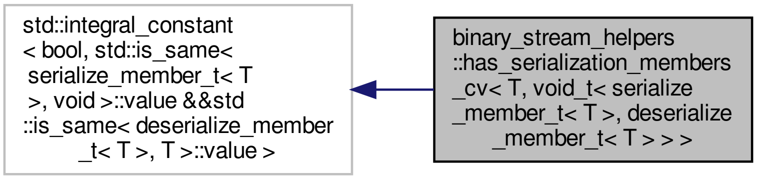
6.8 binary_stream_helpers::has_serialization_members_cv< T, void_t< serialize_member_t< T >, deserialize_member_t< T > > > Struct Template Reference

```
#include <binarystream.h>
```

Inheritance diagram for binary_stream_helpers::has_serialization_members_cv< T, void_t< serialize_member_t< T >, deserialize_member_t< T > > >:


std::integral_constant < bool, std::is_same< serialize_member_t< T >, void >::value &&std ::is_same< deserialize_member _t< T >, T >::value >
binary_stream_helpers ::has_serialization_members _cv< T, void_t< serialize _member_t< T >, deserialize _member_t< T > > >

Collaboration diagram for binary_stream_helpers::has_serialization_members_cv< T, void_t< serialize_member_t< T >, deserialize_member_t< T > > >:


std::integral_constant < bool, std::is_same< serialize_member_t< T >, void >::value &&std ::is_same< deserialize_member _t< T >, T >::value >
binary_stream_helpers ::has_serialization_members _cv< T, void_t< serialize _member_t< T >, deserialize _member_t< T > > >

6.8.1 Detailed Description

template<typename T>
struct binary_stream_helpers::has_serialization_members_cv< T, void_t< serialize_member_t< T >, deserialize_member_t< T > > >

Definition at line 122 of file binarystream.h.

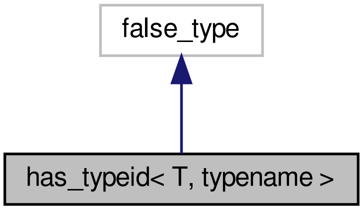
The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/binarystream.h

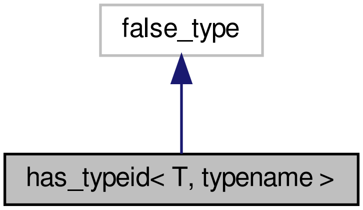
6.9 has_typeid< T, typename > Struct Template Reference

```
#include <typeid.h>
```

Inheritance diagram for has_typeid< T, typename >:



Collaboration diagram for has_typeid< T, typename >:



6.9.1 Detailed Description

template<typename T, typename = void>
struct has_typeid< T, typename >

Definition at line 180 of file typeid.h.

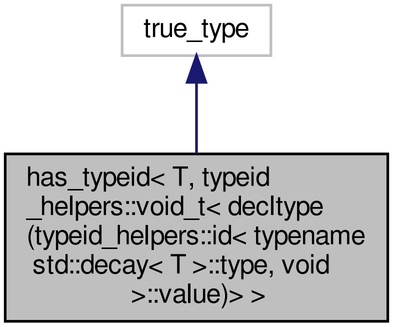
The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

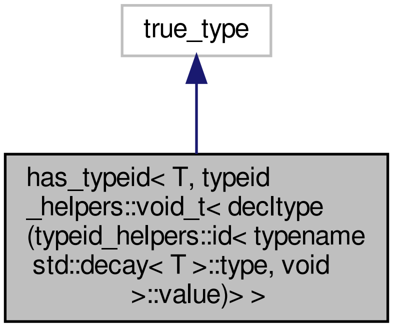
6.10 has_typeid< T, typeid_helpers::void_t< decltype(typeid_helpers::id< typename std↩ ::decay< T >::type, void >::value)> > Struct Template Reference

```
#include <typeid.h>
```

Inheritance diagram for has_typeid< T, typeid_helpers::void_t< decltype(typeid_helpers::id< typename std::decay< T >::type, void >::value)> >:


true_type
has_typeid< T, typeid _helpers::void_t< decltype (typeid_helpers::id< typename std::decay< T >::type, void >::value)> >

Collaboration diagram for has_typeid< T, typeid_helpers::void_t< decltype(typeid_helpers::id< typename std::decay< T >::type, void >::value)> >:


true_type
has_typeid< T, typeid _helpers::void_t< decltype (typeid_helpers::id< typename std::decay< T >::type, void >::value)> >

6.10.1 Detailed Description

template<typename T>
struct has_typeid< T, typeid_helpers::void_t< decltype(typeid_helpers::id< typename std::decay< T >::type, void >::value)> >

Definition at line 183 of file typeid.h.

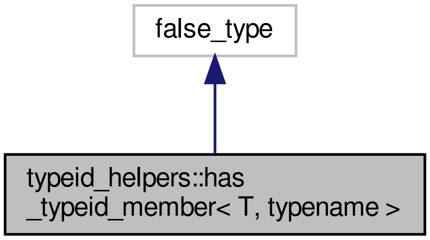
The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

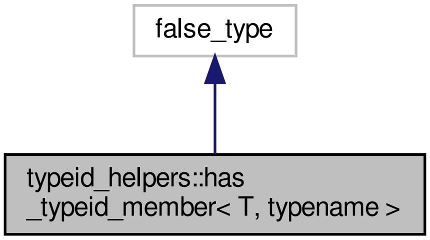
6.11 typeid_helpers::has_typeid_member< T, typename > Struct Template Reference

```
#include <typeid.h>
```

Inheritance diagram for typeid_helpers::has_typeid_member< T, typename >:



Collaboration diagram for typeid_helpers::has_typeid_member< T, typename >:



6.11.1 Detailed Description

template<typename T, typename = void>
struct typeid_helpers::has_typeid_member< T, typename >

Definition at line 33 of file typeid.h.

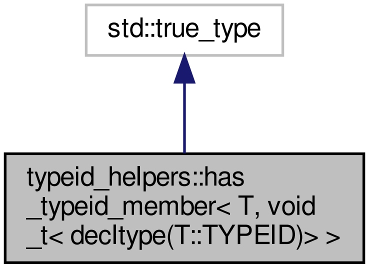
The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

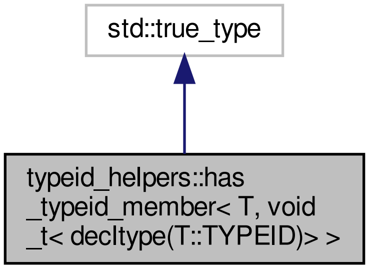
6.12 typeid_helpers::has_typeid_member< T, void_t< decltype(T::TYPEID)> > Struct Template Reference

```
#include <typeid.h>
```

Inheritance diagram for typeid_helpers::has_typeid_member< T, void_t< decltype(T::TYPEID)> >:



Collaboration diagram for typeid_helpers::has_typeid_member< T, void_t< decltype(T::TYPEID)> >:



6.12.1 Detailed Description

template<typename T>
struct typeid_helpers::has_typeid_member< T, void_t< decltype(T::TYPEID)> >

Definition at line 36 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.13 typeid_helpers::id< T, V > Struct Template Reference

```
#include <typeid.h>
```

6.13.1 Detailed Description

template<typename T, typename V = void>
struct typeid_helpers::id< T, V >

Definition at line 47 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.14 typeid_helpers::id< double, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -10

6.14.1 Detailed Description

template<>
struct typeid_helpers::id< double, void >

Definition at line 106 of file typeid.h.

6.14.2 Member Data Documentation

6.14.2.1 value

```
const int32_t typeid_helpers::id< double, void >::value = -10  [static]
```

Definition at line 107 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.15 typeid_helpers::id< float, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -9

6.15.1 Detailed Description

template<>
struct typeid_helpers::id< float, void >

Definition at line 101 of file typeid.h.

6.15.2 Member Data Documentation

6.15.2.1 value

```
const int32_t typeid_helpers::id< float, void >::value = -9  [static]
```

Definition at line 102 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.16 typeid_helpers::id< int16_t, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -3

6.16.1 Detailed Description

template<>
struct typeid_helpers::id< int16_t, void >

Definition at line 71 of file typeid.h.

6.16.2 Member Data Documentation

6.16.2.1 value

```
const int32_t typeid_helpers::id< int16_t, void >::value = -3  [static]
```

Definition at line 72 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.17 typeid_helpers::id< int32_t, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -5

6.17.1 Detailed Description

template<>
struct typeid_helpers::id< int32_t, void >

Definition at line 81 of file typeid.h.

6.17.2 Member Data Documentation

6.17.2.1 value

```
const int32_t typeid_helpers::id< int32_t, void >::value = -5  [static]
```

Definition at line 82 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.18 typeid_helpers::id< int64_t, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -7

6.18.1 Detailed Description

template<>
struct typeid_helpers::id< int64_t, void >

Definition at line 91 of file typeid.h.

6.18.2 Member Data Documentation

6.18.2.1 value

```
const int32_t typeid_helpers::id< int64_t, void >::value = -7  [static]
```

Definition at line 92 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.19 typeid_helpers::id< int8_t, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -1

6.19.1 Detailed Description

template<>
struct typeid_helpers::id< int8_t, void >

Definition at line 61 of file typeid.h.

6.19.2 Member Data Documentation

6.19.2.1 value

```
const int32_t typeid_helpers::id< int8_t, void >::value = -1  [static]
```

Definition at line 62 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.20 typeid_helpers::id< std::string, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -11

6.20.1 Detailed Description

template<>
struct typeid_helpers::id< std::string, void >

Definition at line 111 of file typeid.h.

6.20.2 Member Data Documentation

6.20.2.1 value

```
const int32_t typeid_helpers::id< std::string, void >::value = -11  [static]
```

Definition at line 112 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.21 typeid_helpers::id< std::vector< double >, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -30

6.21.1 Detailed Description

template<>
struct typeid_helpers::id< std::vector< double >, void >

Definition at line 163 of file typeid.h.

6.21.2 Member Data Documentation

6.21.2.1 value

```
const int32_t typeid_helpers::id< std::vector< double >, void >::value = -30  [static]
```

Definition at line 164 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.22 typeid_helpers::id< std::vector< float >, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -29

6.22.1 Detailed Description

template<>
struct typeid_helpers::id< std::vector< float >, void >

Definition at line 158 of file typeid.h.

6.22.2 Member Data Documentation

6.22.2.1 value

```
const int32_t typeid_helpers::id< std::vector< float >, void >::value = -29  [static]
```

Definition at line 159 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.23 typeid_helpers::id< std::vector< int16_t >, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -23

6.23.1 Detailed Description

template<>
struct typeid_helpers::id< std::vector< int16_t >, void >

Definition at line 128 of file typeid.h.

6.23.2 Member Data Documentation

6.23.2.1 value

```
const int32_t typeid_helpers::id< std::vector< int16_t >, void >::value = -23  [static]
```

Definition at line 129 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.24 typeid_helpers::id< std::vector< int32_t >, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -25

6.24.1 Detailed Description

template<>
struct typeid_helpers::id< std::vector< int32_t >, void >

Definition at line 138 of file typeid.h.

6.24.2 Member Data Documentation

6.24.2.1 value

```
const int32_t typeid_helpers::id< std::vector< int32_t >, void >::value = -25  [static]
```

Definition at line 139 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.25 typeid_helpers::id< std::vector< int64_t >, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -27

6.25.1 Detailed Description

template<>
struct typeid_helpers::id< std::vector< int64_t >, void >

Definition at line 148 of file typeid.h.

6.25.2 Member Data Documentation

6.25.2.1 value

```
const int32_t typeid_helpers::id< std::vector< int64_t >, void >::value = -27  [static]
```

Definition at line 149 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.26 typeid_helpers::id< std::vector< int8_t >, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -21

6.26.1 Detailed Description

template<>
struct typeid_helpers::id< std::vector< int8_t >, void >

Definition at line 118 of file typeid.h.

6.26.2 Member Data Documentation

6.26.2.1 value

```
const int32_t typeid_helpers::id< std::vector< int8_t >, void >::value = -21  [static]
```

Definition at line 119 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.27 typeid_helpers::id< std::vector< std::string >, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -31

6.27.1 Detailed Description

template<>
struct typeid_helpers::id< std::vector< std::string >, void >

Definition at line 168 of file typeid.h.

6.27.2 Member Data Documentation

6.27.2.1 value

```
const int32_t typeid_helpers::id< std::vector< std::string >, void >::value = -31  [static]
```

Definition at line 169 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.28 typeid_helpers::id< std::vector< uint16_t >, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -24

6.28.1 Detailed Description

template<>
struct typeid_helpers::id< std::vector< uint16_t >, void >

Definition at line 133 of file typeid.h.

6.28.2 Member Data Documentation

6.28.2.1 value

```
const int32_t typeid_helpers::id< std::vector< uint16_t >, void >::value = -24  [static]
```

Definition at line 134 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.29 typeid_helpers::id< std::vector< uint32_t >, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -26

6.29.1 Detailed Description

template<>
struct typeid_helpers::id< std::vector< uint32_t >, void >

Definition at line 143 of file typeid.h.

6.29.2 Member Data Documentation

6.29.2.1 value

```
const int32_t typeid_helpers::id< std::vector< uint32_t >, void >::value = -26  [static]
```

Definition at line 144 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.30 typeid_helpers::id< std::vector< uint64_t >, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -28

6.30.1 Detailed Description

template<>
struct typeid_helpers::id< std::vector< uint64_t >, void >

Definition at line 153 of file typeid.h.

6.30.2 Member Data Documentation

6.30.2.1 value

```
const int32_t typeid_helpers::id< std::vector< uint64_t >, void >::value = -28 [static]
```

Definition at line 154 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.31 typeid_helpers::id< std::vector< uint8_t >, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -22

6.31.1 Detailed Description

template<>
struct typeid_helpers::id< std::vector< uint8_t >, void >

Definition at line 123 of file typeid.h.

6.31.2 Member Data Documentation

6.31.2.1 value

```
const int32_t typeid_helpers::id< std::vector< uint8_t >, void >::value = -22 [static]
```

Definition at line 124 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.32 typeid_helpers::id< T, typename std::enable_if< has_typeid_member< T >::value >::type > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = T::TYPEID

6.32.1 Detailed Description

template<typename T>
struct typeid_helpers::id< T, typename std::enable_if< has_typeid_member< T >::value >::type >

Definition at line 51 of file typeid.h.

6.32.2 Member Data Documentation

6.32.2.1 value

```
template<typename T >
const int32_t typeid_helpers::id< T, typename std::enable_if< has_typeid_member< T >::value >↩
::type >::value = T::TYPEID  [static]
```

Definition at line 52 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.33 typeid_helpers::id< uint16_t, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -4

6.33.1 Detailed Description

template<>
struct typeid_helpers::id< uint16_t, void >

Definition at line 76 of file typeid.h.

6.33.2 Member Data Documentation

6.33.2.1 value

```
const int32_t typeid_helpers::id< uint16_t, void >::value = -4  [static]
```

Definition at line 77 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.34 typeid_helpers::id< uint32_t, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -6

6.34.1 Detailed Description

template<>
struct typeid_helpers::id< uint32_t, void >

Definition at line 86 of file typeid.h.

6.34.2 Member Data Documentation

6.34.2.1 value

```
const int32_t typeid_helpers::id< uint32_t, void >::value = -6  [static]
```

Definition at line 87 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.35 typeid_helpers::id< uint64_t, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -8

6.35.1 Detailed Description

template<>
struct typeid_helpers::id< uint64_t, void >

Definition at line 96 of file typeid.h.

6.35.2 Member Data Documentation

6.35.2.1 value

```
const int32_t typeid_helpers::id< uint64_t, void >::value = -8  [static]
```

Definition at line 97 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.36 typeid_helpers::id< uint8_t, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = -2

6.36.1 Detailed Description

template<>
struct typeid_helpers::id< uint8_t, void >

Definition at line 66 of file typeid.h.

6.36.2 Member Data Documentation

6.36.2.1 value

```
const int32_t typeid_helpers::id< uint8_t, void >::value = -2  [static]
```

Definition at line 67 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.37 typeid_helpers::id< void, void > Struct Template Reference

```
#include <typeid.h>
```

Static Public Attributes

- static const int32_t value = 0

6.37.1 Detailed Description

template<>
struct typeid_helpers::id< void, void >

Definition at line 56 of file typeid.h.

6.37.2 Member Data Documentation

6.37.2.1 value

```
const int32_t typeid_helpers::id< void, void >::value = 0  [static]
```

Definition at line 57 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.38 MemBuf Class Reference

```
#include <MemBuf.h>
```

Public Member Functions

- MemBuf (int sizebytes=1024)
- ∼MemBuf ()
- void resize ()
- void rewind ()
- void add (const std::string &)
- const char ∗ getBufPtr ()

Private Attributes

- std::string buffer

6.38.1 Detailed Description

Definition at line 23 of file MemBuf.h.

6.38.2 Constructor & Destructor Documentation

6.38.2.1 MemBuf()

```
MemBuf::MemBuf (
             int sizebytes = 1024 )
```

Definition at line 34 of file MemBuf.cpp.

References buffer.

6.38.2.2 ∼MemBuf()

```
MemBuf::∼MemBuf ( )
```

Definition at line 32 of file MemBuf.cpp.

6.38.3 Member Function Documentation

6.38.3.1 add()

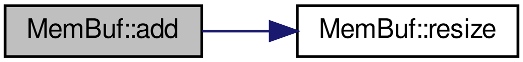
```
void MemBuf::add (
             const std::string & buf )
```

Definition at line 46 of file MemBuf.cpp.

References buffer, and resize().

Referenced by RInside::parseEval().

Here is the call graph for this function:



6.38.3.2 getBufPtr()

```
const char* MemBuf::getBufPtr ( )  [inline]
```

Definition at line 33 of file MemBuf.h.

Referenced by RInside::parseEval().

6.38.3.3 resize()

```
void MemBuf::resize ( )
```

Definition at line 38 of file MemBuf.cpp.

References buffer.

Referenced by add().

6.38.3.4 rewind()

```
void MemBuf::rewind ( )
```

Definition at line 42 of file MemBuf.cpp.

References buffer.

Referenced by RInside::parseEval().

6.38.4 Member Data Documentation

6.38.4.1 buffer

```
std::string MemBuf::buffer  [private]
```

Definition at line 25 of file MemBuf.h.

Referenced by add(), MemBuf(), resize(), and rewind().

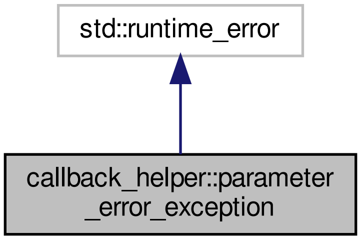
The documentation for this class was generated from the following files:

- inst/include/MemBuf.h
- src/MemBuf.cpp

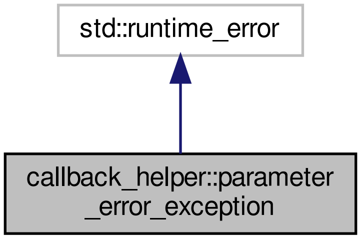
6.39 callback_helper::parameter_error_exception Class Reference

```
#include <callback_helper.h>
```

Inheritance diagram for callback_helper::parameter_error_exception:



Collaboration diagram for callback_helper::parameter_error_exception:



Public Member Functions

- parameter_error_exception (const std::string &error)

6.39.1 Detailed Description

Definition at line 31 of file callback_helper.h.

6.39.2 Constructor & Destructor Documentation

6.39.2.1 parameter_error_exception()

```
callback_helper::parameter_error_exception::parameter_error_exception (
             const std::string & error )  [inline], [explicit]
```

Definition at line 33 of file callback_helper.h.

The documentation for this class was generated from the following file:

- inst/examples/sandboxed_server/client/callback_helper.h

6.40 Planet Struct Reference

Public Attributes

- double x
- double y
- double m
- double vx
- double vy

6.40.1 Detailed Description

Definition at line 16 of file rinside_interactive0.cpp.

6.40.2 Member Data Documentation

6.40.2.1 m

```
double Planet::m
```

Definition at line 18 of file rinside_interactive0.cpp.

Referenced by Wrapper::getData(), and Wrapper::setData().

6.40.2.2 vx

```
double Planet::vx
```

Definition at line 19 of file rinside_interactive0.cpp.

Referenced by Wrapper::getData(), and Wrapper::setData().

6.40.2.3 vy

```
double Planet::vy
```

Definition at line 19 of file rinside_interactive0.cpp.

Referenced by Wrapper::getData(), and Wrapper::setData().

6.40.2.4 x

```
double Planet::x
```

Definition at line 17 of file rinside_interactive0.cpp.

Referenced by Wrapper::getData(), Solver::Iteration(), and Wrapper::setData().

6.40.2.5 y

```
double Planet::y
```

Definition at line 17 of file rinside_interactive0.cpp.

Referenced by Wrapper::getData(), Solver::Iteration(), and Wrapper::setData().

The documentation for this struct was generated from the following file:

- inst/examples/standard/rinside_interactive0.cpp

6.41 RInside::Proxy Class Reference

```
#include <RInside.h>
```

Public Member Functions

- Proxy (SEXP xx)
- template<typename T >
  operator T ()

Private Attributes

- Rcpp::RObject x

6.41.1 Detailed Description

Definition at line 59 of file RInside.h.

6.41.2 Constructor & Destructor Documentation

6.41.2.1 Proxy()

```
RInside::Proxy::Proxy (
             SEXP xx )  [inline]
```

Definition at line 61 of file RInside.h.

6.41.3 Member Function Documentation

6.41.3.1 operator T()

```
template<typename T >
RInside::Proxy::operator T ( )  [inline]
```

Definition at line 64 of file RInside.h.

References x.

6.41.4 Member Data Documentation

6.41.4.1 x

```
Rcpp::RObject RInside::Proxy::x  [private]
```

Definition at line 68 of file RInside.h.

Referenced by operator T().

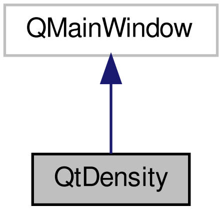
The documentation for this class was generated from the following file:

- inst/include/RInside.h

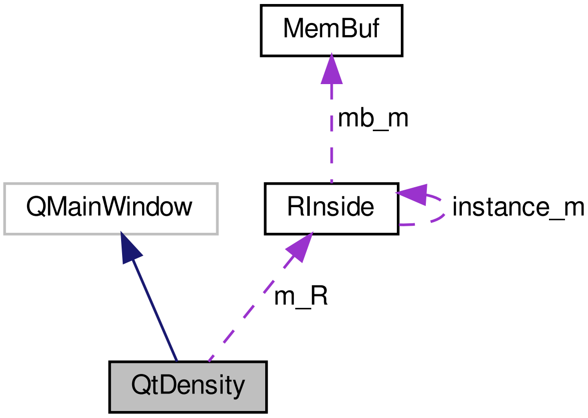
6.42 QtDensity Class Reference

```
#include <qtdensity.h>
```

Inheritance diagram for QtDensity:



Collaboration diagram for QtDensity:



Public Member Functions

- QtDensity (RInside &R)

Private Slots

- void getBandwidth (int bw)
- void getKernel (int kernel)
- void getRandomDataCmd (QString txt)
- void runRandomDataCmd (void)

Private Member Functions

- void setupDisplay (void)
- void plot (void)
- void filterFile (void)

Private Attributes

- QSvgWidget ∗ m_svg
- RInside & m_R
- QString m_tempfile
- QString m_svgfile
- int m_bw
- int m_kernel
- QString m_cmd

6.42.1 Detailed Description

Definition at line 28 of file qtdensity.h.

6.42.2 Constructor & Destructor Documentation

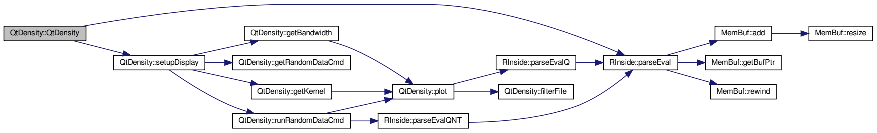
6.42.2.1 QtDensity()

```
QtDensity::QtDensity (
            RInside & R )
```

Definition at line 10 of file qtdensity.cpp.

References m_bw, m_cmd, m_kernel, m_R, m_svgfile, m_tempfile, RInside::parseEval(), and setupDisplay().

Here is the call graph for this function:



6.42.3 Member Function Documentation

6.42.3.1 filterFile()

```
void QtDensity::filterFile (
             void  )  [private]
```

Definition at line 131 of file qtdensity.cpp.

References m_svgfile, and m_tempfile.

Referenced by plot().

6.42.3.2 getBandwidth

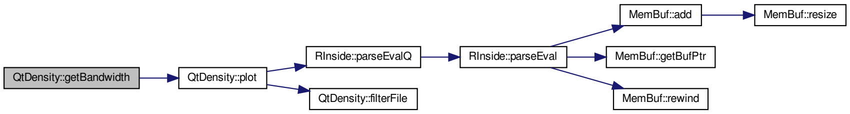
```
void QtDensity::getBandwidth (
             int bw )  [private], [slot]
```

Definition at line 107 of file qtdensity.cpp.

References m_bw, and plot().

Referenced by setupDisplay().

Here is the call graph for this function:



6.42.3.3 getKernel

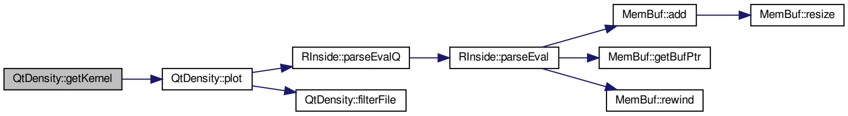
```
void QtDensity::getKernel (
             int kernel )  [private], [slot]
```

Definition at line 114 of file qtdensity.cpp.

References m_kernel, and plot().

Referenced by setupDisplay().

Here is the call graph for this function:



6.42.3.4 getRandomDataCmd

```
void QtDensity::getRandomDataCmd (
             QString txt )  [private], [slot]
```

Definition at line 121 of file qtdensity.cpp.

References m_cmd.

Referenced by setupDisplay().

6.42.3.5 plot()

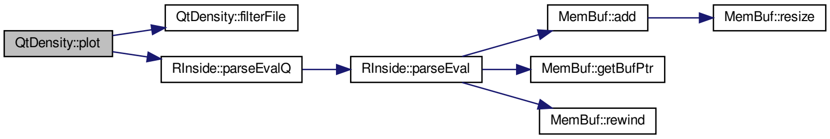
```
void QtDensity::plot (
             void  )  [private]
```

Definition at line 94 of file qtdensity.cpp.

References filterFile(), m_bw, m_kernel, m_R, m_svg, m_svgfile, and RInside::parseEvalQ().

Referenced by getBandwidth(), getKernel(), and runRandomDataCmd().

Here is the call graph for this function:



6.42.3.6 runRandomDataCmd

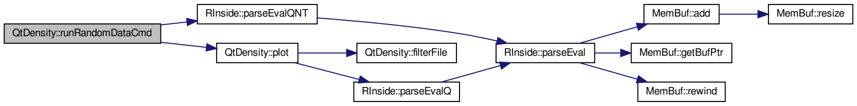
```
void QtDensity::runRandomDataCmd (
             void  )  [private], [slot]
```

Definition at line 125 of file qtdensity.cpp.

References m_cmd, m_R, RInside::parseEvalQNT(), and plot().

Referenced by setupDisplay().

Here is the call graph for this function:



6.42.3.7 setupDisplay()

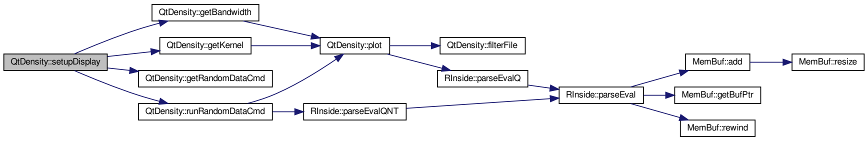
```
void QtDensity::setupDisplay (
             void  )  [private]
```

Definition at line 21 of file qtdensity.cpp.

References getBandwidth(), getKernel(), getRandomDataCmd(), m_bw, m_cmd, m_svg, and runRandomDataCmd().

Referenced by QtDensity().

Here is the call graph for this function:



6.42.4 Member Data Documentation

6.42.4.1 m_bw

```
int QtDensity::m_bw [private]
```

Definition at line 50 of file qtdensity.h.

Referenced by getBandwidth(), plot(), QtDensity(), and setupDisplay().

6.42.4.2 m_cmd

```
QString QtDensity::m_cmd [private]
```

Definition at line 51 of file qtdensity.h.

Referenced by getRandomDataCmd(), QtDensity(), runRandomDataCmd(), and setupDisplay().

6.42.4.3 m_kernel

```
int QtDensity::m_kernel [private]
```

Definition at line 50 of file qtdensity.h.

Referenced by getKernel(), plot(), and QtDensity().

6.42.4.4 m_R

```
RInside& QtDensity::m_R [private]
```

Definition at line 47 of file qtdensity.h.

Referenced by plot(), QtDensity(), and runRandomDataCmd().

6.42.4.5 m_svg

```
QSvgWidget* QtDensity::m_svg [private]
```

Definition at line 46 of file qtdensity.h.

Referenced by plot(), and setupDisplay().

6.42.4.6 m_svgfile

```
QString QtDensity::m_svgfile  [private]
```

Definition at line 49 of file qtdensity.h.

Referenced by filterFile(), plot(), and QtDensity().

6.42.4.7 m_tempfile

```
QString QtDensity::m_tempfile  [private]
```

Definition at line 48 of file qtdensity.h.

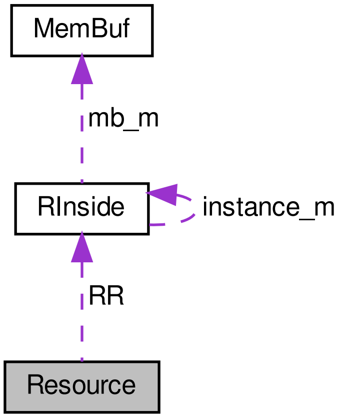
Referenced by filterFile(), and QtDensity().

The documentation for this class was generated from the following files:

- inst/examples/qt/qtdensity.h
- inst/examples/qt/qtdensity.cpp

6.43 Resource Class Reference

Collaboration diagram for Resource:



Public Member Functions

- Resource ()
- void use ()
- int getValue ()

Private Attributes

- int i
- RInside & RR
- boost::mutex guard

6.43.1 Detailed Description

Definition at line 8 of file boostEx.cpp.

6.43.2 Constructor & Destructor Documentation

6.43.2.1 Resource()

```
Resource::Resource ( )  [inline]
```

Definition at line 10 of file boostEx.cpp.

6.43.3 Member Function Documentation

6.43.3.1 getValue()

```
int Resource::getValue ( )  [inline]
```

Definition at line 17 of file boostEx.cpp.

References i.

Referenced by main().

6.43.3.2 use()

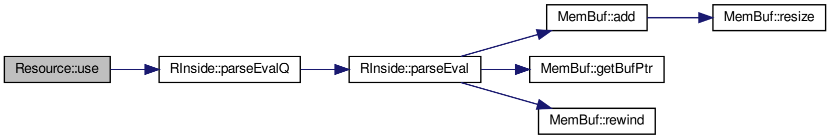
```
void Resource::use ( ) [inline]
```

Definition at line 12 of file boostEx.cpp.

References guard, i, RInside::parseEvalQ(), and RR.

Referenced by thread_func().

Here is the call graph for this function:



6.43.4 Member Data Documentation

6.43.4.1 guard

```
boost::mutex Resource::guard [private]
```

Definition at line 22 of file boostEx.cpp.

Referenced by use().

6.43.4.2 i

```
int Resource::i [private]
```

Definition at line 20 of file boostEx.cpp.

Referenced by getValue(), and use().

6.43.4.3 RR

```
RInside& Resource::RR  [private]
```

Definition at line 21 of file boostEx.cpp.

Referenced by use().

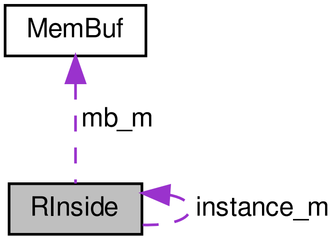
The documentation for this class was generated from the following file:

- inst/examples/threads/boostEx.cpp

6.44 RInside Class Reference

```
#include <RInside.h>
```

Collaboration diagram for RInside:



Classes

- class Proxy

Public Member Functions

- int parseEval (const std::string &line, SEXP &ans)
- void parseEvalQ (const std::string &line)
- void parseEvalQNT (const std::string &line)
- Proxy parseEval (const std::string &line)
- Proxy parseEvalNT (const std::string &line)
- template<typename T >
  void assign (const T &object, const std::string &nam)
- RInside ()
- RInside (const int argc, const char ∗const argv[ ], const bool loadRcpp=true, const bool verbose=false, const bool interactive=false)
- ∼RInside ()
- void setVerbose (const bool verbose)
- Rcpp::Environment::Binding operator[ ] (const std::string &name)
- void repl ()

Static Public Member Functions

- static RInside & instance ()
- static RInside ∗ instancePtr ()

Private Member Functions

- void init_tempdir (void)
- void init_rand (void)
- void autoloads (void)
- void initialize (const int argc, const char ∗const argv[ ], const bool loadRcpp, const bool verbose, const bool interactive)

Private Attributes

- MemBuf mb_m
- Rcpp::Environment ∗ global_env_m
- bool verbose_m
- bool interactive_m

Static Private Attributes

- static RInside ∗ instance_m = 0

6.44.1 Detailed Description

Definition at line 29 of file RInside.h.

6.44.2 Constructor & Destructor Documentation

6.44.2.1 RInside() [1/2]

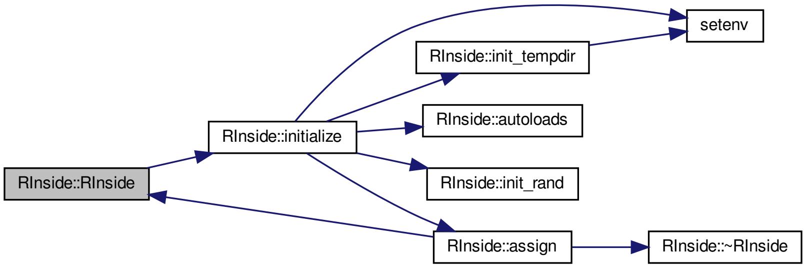
```
RInside::RInside ( )
```

Definition at line 56 of file RInside.cpp.

References initialize().

Referenced by assign().

Here is the call graph for this function:



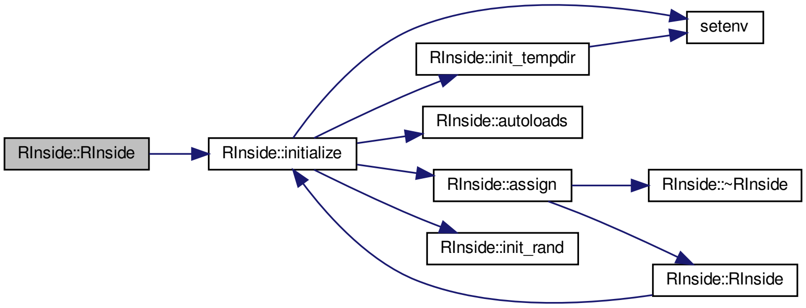
6.44.2.2 RInside() [2/2]

```
RInside::RInside (
            const int argc,
            const char *const argv[],
            const bool loadRcpp = true,
            const bool verbose = false,
            const bool interactive = false )
```

Definition at line 98 of file RInside.cpp.

References initialize().

Here is the call graph for this function:



6.44.2.3 ∼RInside()

```
RInside::∼RInside ( )
```

Definition at line 43 of file RInside.cpp.

References global_env_m, and instance_m.

Referenced by assign().

6.44.3 Member Function Documentation

6.44.3.1 assign()

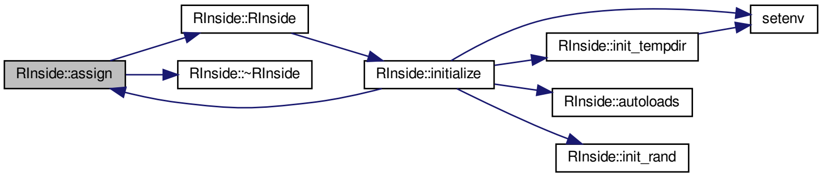
```
template<typename T >
void RInside::assign (
             const T & object,
             const std::string & nam )  [inline]
```

Definition at line 78 of file RInside.h.

References RInside(), verbose, and ∼RInside().

Referenced by initialize(), main(), and passToR().

Here is the call graph for this function:



6.44.3.2 autoloads()

```
void RInside::autoloads (
             void  )  [private]
```

Definition at line 239 of file RInside.cpp.

References global_env_m.

Referenced by initialize().

6.44.3.3 init_rand()

```
void RInside::init_rand (
             void  )  [private]
```

Definition at line 230 of file RInside.cpp.

Referenced by initialize().

6.44.3.4 init_tempdir()

```
void RInside::init_tempdir (
             void  )  [private]
```

Definition at line 212 of file RInside.cpp.

References setenv().

Referenced by initialize().

Here is the call graph for this function:

RInside::init_tempdir
setenv

6.44.3.5 initialize()

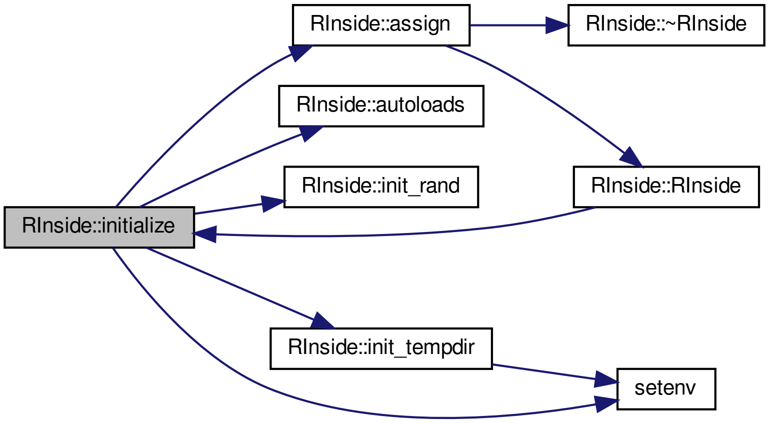
```
void RInside::initialize (
            const int argc,
            const char *const argv[],
            const bool loadRcpp,
            const bool verbose,
            const bool interactive )  [private]
```

Definition at line 108 of file RInside.cpp.

References assign(), autoloads(), global_env_m, init_rand(), init_tempdir(), instance_m, interactive_m, programName, R_CStackLimit, setenv(), verbose, and verbose_m.

Referenced by RInside().

Here is the call graph for this function:



6.44.3.6 instance()

```
RInside & RInside::instance ( )  [static]
```

Definition at line 416 of file RInside.cpp.

References instance_m.

Referenced by createApplication(), repl(), and setVerbose().

6.44.3.7 instancePtr()

```
RInside * RInside::instancePtr ( )  [static]
```

Definition at line 420 of file RInside.cpp.

References instance_m.

Referenced by setVerbose().

6.44.3.8 operator[]()

```
Rcpp::Environment::Binding RInside::operator[] (
             const std::string & name )
```

Definition at line 412 of file RInside.cpp.

References global_env_m.

Referenced by setVerbose().

6.44.3.9 parseEval() [1/2]

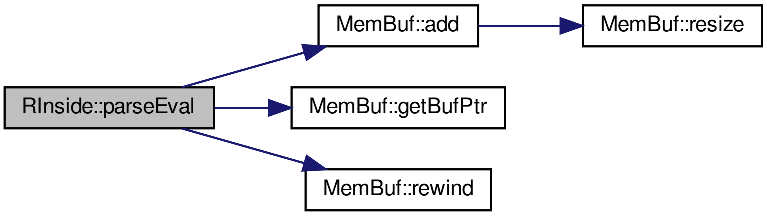
```
int RInside::parseEval (
             const std::string & line,
             SEXP & ans )
```

Definition at line 326 of file RInside.cpp.

References MemBuf::add(), MemBuf::getBufPtr(), global_env_m, mb_m, programName, MemBuf::rewind(), and verbose_m.

Referenced by DensityApp::DensityApp(), do_work(), evalInR(), main(), parseEval(), parseEvalNT(), parseEvalQ(), parseEvalQNT(), QtDensity::QtDensity(), and RInsideServer::run().

Here is the call graph for this function:



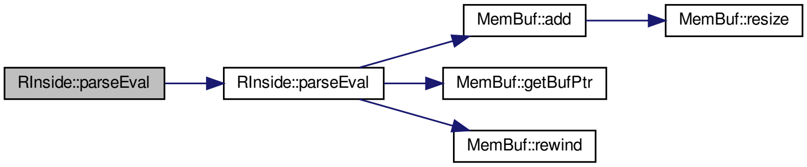
6.44.3.10 parseEval() [2/2]

```
RInside::Proxy RInside::parseEval (
            const std::string & line )
```

Definition at line 397 of file RInside.cpp.

References parseEval().

Here is the call graph for this function:



6.44.3.11 parseEvalNT()

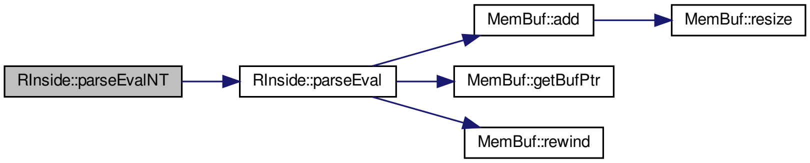
```
RInside::Proxy RInside::parseEvalNT (
            const std::string & line )
```

Definition at line 406 of file RInside.cpp.

References parseEval().

Referenced by main().

Here is the call graph for this function:



6.44.3.12 parseEvalQ()

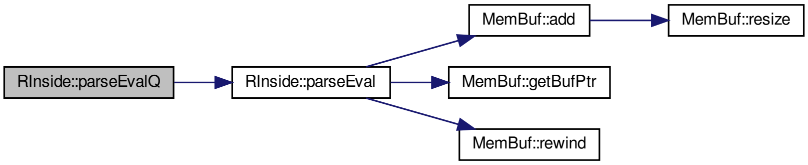
```
void RInside::parseEvalQ (
             const std::string & line )
```

Definition at line 384 of file RInside.cpp.

References parseEval().

Referenced by evalQuietlyInR(), initalize(), main(), QtDensity::plot(), DensityApp::plot(), RInsideServer::run(), and Resource::use().

Here is the call graph for this function:



6.44.3.13 parseEvalQNT()

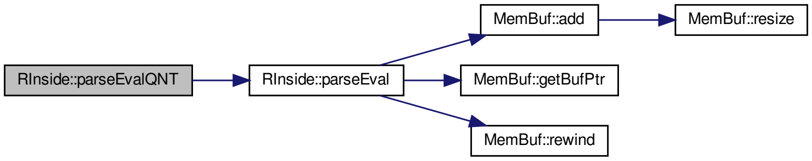
```
void RInside::parseEvalQNT (
             const std::string & line )
```

Definition at line 392 of file RInside.cpp.

References parseEval().

Referenced by main(), DensityApp::reportEdit(), and QtDensity::runRandomDataCmd().

Here is the call graph for this function:



6.44.3.14 repl()

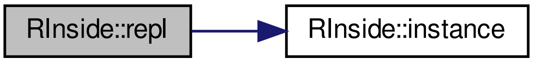
```
void RInside::repl ( )
```

Definition at line 424 of file RInside.cpp.

References instance().

Referenced by main(), and setVerbose().

Here is the call graph for this function:



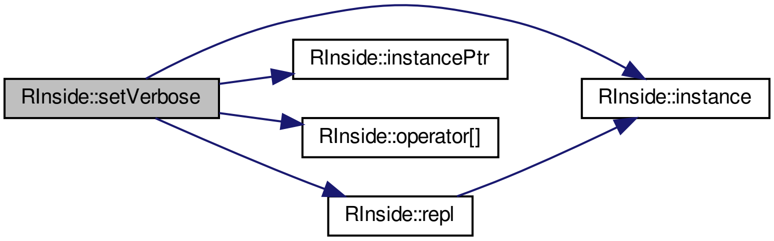
6.44.3.15 setVerbose()

```
void RInside::setVerbose (
            const bool verbose )  [inline]
```

Definition at line 88 of file RInside.h.

References instance(), instancePtr(), operator[ ](), repl(), and verbose.

Here is the call graph for this function:



6.44.4 Member Data Documentation

6.44.4.1 global_env_m

```
Rcpp::Environment* RInside::global_env_m  [private]
```

Definition at line 32 of file RInside.h.

Referenced by autoloads(), initialize(), operator[ ](), parseEval(), and ∼RInside().

6.44.4.2 instance_m

```
RInside * RInside::instance_m = 0  [static], [private]
```

Definition at line 44 of file RInside.h.

Referenced by initialize(), instance(), instancePtr(), and ∼RInside().

6.44.4.3 interactive_m

```
bool RInside::interactive_m  [private]
```

Definition at line 35 of file RInside.h.

Referenced by initialize().

6.44.4.4 mb_m

```
MemBuf RInside::mb_m  [private]
```

Definition at line 31 of file RInside.h.

Referenced by parseEval().

6.44.4.5 verbose_m

```
bool RInside::verbose_m  [private]
```

Definition at line 34 of file RInside.h.

Referenced by initialize(), and parseEval().

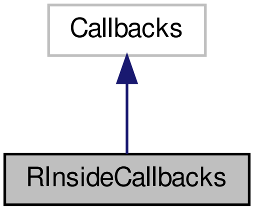
The documentation for this class was generated from the following files:

- inst/include/RInside.h
- src/RInside.cpp

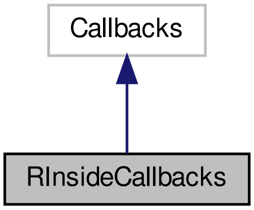
6.45 RInsideCallbacks Class Reference

```
#include <rinside_callbacks.h>
```

Inheritance diagram for RInsideCallbacks:



Collaboration diagram for RInsideCallbacks:



Public Member Functions

- virtual std::string ReadConsole (const char ∗prompt, bool addtohistory)
- virtual void WriteConsole (const std::string &line, int type)
- virtual void FlushConsole ()
- virtual void ResetConsole ()
- virtual void CleanerrConsole ()
- virtual void Busy (bool)
- virtual void ShowMessage (const char ∗message)
- virtual void Suicide (const char ∗message)
- virtual bool has_ReadConsole ()
- virtual bool has_WriteConsole ()
- virtual bool has_FlushConsole ()
- virtual bool has_ResetConsole ()
- virtual bool has_CleanerrConsole ()
- virtual bool has_Busy ()
- virtual bool has_ShowMessage ()
- virtual bool has_Suicide ()
- void resetConsoleOutput ()
- std::string getConsoleOutput ()

Private Attributes

- std::ostringstream output_buffer

6.45.1 Detailed Description

Definition at line 1 of file rinside_callbacks.h.

6.45.2 Member Function Documentation

6.45.2.1 Busy()

```
virtual void RInsideCallbacks::Busy (
             bool  )  [inline], [virtual]
```

Definition at line 22 of file rinside_callbacks.h.

6.45.2.2 CleanerrConsole()

```
virtual void RInsideCallbacks::CleanerrConsole ( )  [inline], [virtual]
```

Definition at line 19 of file rinside_callbacks.h.

6.45.2.3 FlushConsole()

```
virtual void RInsideCallbacks::FlushConsole ( )  [inline], [virtual]
```

Definition at line 13 of file rinside_callbacks.h.

6.45.2.4 getConsoleOutput()

```
std::string RInsideCallbacks::getConsoleOutput ( )  [inline]
```

Definition at line 49 of file rinside_callbacks.h.

References output_buffer.

Referenced by main(), and RInsideServer::run().

6.45.2.5 has_Busy()

```
virtual bool RInsideCallbacks::has_Busy ( )  [inline], [virtual]
```

Definition at line 40 of file rinside_callbacks.h.

6.45.2.6 has_CleanerrConsole()

```
virtual bool RInsideCallbacks::has_CleanerrConsole ( )  [inline], [virtual]
```

Definition at line 39 of file rinside_callbacks.h.

6.45.2.7 has_FlushConsole()

```
virtual bool RInsideCallbacks::has_FlushConsole ( )  [inline], [virtual]
```

Definition at line 37 of file rinside_callbacks.h.

6.45.2.8 has_ReadConsole()

```
virtual bool RInsideCallbacks::has_ReadConsole ( )  [inline], [virtual]
```

Definition at line 35 of file rinside_callbacks.h.

6.45.2.9 has_ResetConsole()

```
virtual bool RInsideCallbacks::has_ResetConsole ( )  [inline], [virtual]
```

Definition at line 38 of file rinside_callbacks.h.

6.45.2.10 has_ShowMessage()

```
virtual bool RInsideCallbacks::has_ShowMessage ( )  [inline], [virtual]
```

Definition at line 41 of file rinside_callbacks.h.

6.45.2.11 has_Suicide()

```
virtual bool RInsideCallbacks::has_Suicide ( )  [inline], [virtual]
```

Definition at line 42 of file rinside_callbacks.h.

6.45.2.12 has_WriteConsole()

```
virtual bool RInsideCallbacks::has_WriteConsole ( )  [inline], [virtual]
```

Definition at line 36 of file rinside_callbacks.h.

6.45.2.13 ReadConsole()

```
virtual std::string RInsideCallbacks::ReadConsole (
            const char * prompt,
            bool addtohistory )  [inline], [virtual]
```

Definition at line 4 of file rinside_callbacks.h.

6.45.2.14 ResetConsole()

```
virtual void RInsideCallbacks::ResetConsole ( )  [inline], [virtual]
```

Definition at line 16 of file rinside_callbacks.h.

6.45.2.15 resetConsoleOutput()

```
void RInsideCallbacks::resetConsoleOutput ( )  [inline]
```

Definition at line 44 of file rinside_callbacks.h.

References output_buffer.

Referenced by main(), and RInsideServer::run().

6.45.2.16 ShowMessage()

```
virtual void RInsideCallbacks::ShowMessage (
            const char * message )  [inline], [virtual]
```

Definition at line 25 of file rinside_callbacks.h.

6.45.2.17 Suicide()

```
virtual void RInsideCallbacks::Suicide (
            const char * message )  [inline], [virtual]
```

Definition at line 29 of file rinside_callbacks.h.

References LOG.

6.45.2.18 WriteConsole()

```
virtual void RInsideCallbacks::WriteConsole (
             const std::string & line,
             int type )  [inline], [virtual]
```

Definition at line 8 of file rinside_callbacks.h.

References output_buffer.

6.45.3 Member Data Documentation

6.45.3.1 output_buffer

```
std::ostringstream RInsideCallbacks::output_buffer  [private]
```

Definition at line 53 of file rinside_callbacks.h.

Referenced by getConsoleOutput(), resetConsoleOutput(), and WriteConsole().

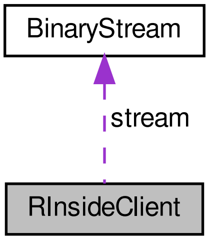
The documentation for this class was generated from the following file:

- inst/examples/sandboxed_server/server/rinside_callbacks.h

6.46 RInsideClient Class Reference

```
#include <rinsideclient.h>
```

Collaboration diagram for RInsideClient:



Public Member Functions

- RInsideClient (BinaryStream &stream)
- ~RInsideClient ()
- void parseEvalQ (const std::string &code)
- template<typename T >
  T parseEval (const std::string &code)
- template<typename T >
  void setValue (const std::string &name, const T &value)
- template<typename T >
  T getValue (const std::string &name)
- template<typename R , typename... Params>
  void setCallback (const std::string &name, std::function< R(Params...)> &callback)
- std::string getConsoleOutput ()
- void initPlot (uint32_t width=800, uint32_t height=600)
- std::string getPlot ()

Private Member Functions

- void runScript (const std::string code, int32_t result_typeid)
- void writeCommand (char command)
- char readReply (bool accept_ok=true, bool accept_value=false)
- void unrecoverable_error (const std::string &error)

Private Attributes

- BinaryStream stream
- uint32_t next_callback_id
- std::map< uint32_t, std::function< void(void)> > callbacks
- bool had_unrecoverable_error
- bool can_send_command

6.46.1 Detailed Description

Definition at line 19 of file rinsideclient.h.

6.46.2 Constructor & Destructor Documentation

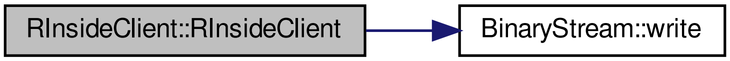
6.46.2.1 RInsideClient()

```
RInsideClient::RInsideClient (
            BinaryStream & stream )
```

Definition at line 12 of file rinsideclient.cpp.

References can_send_command, RIS_MAGIC_NUMBER, stream, and BinaryStream::write().

Here is the call graph for this function:



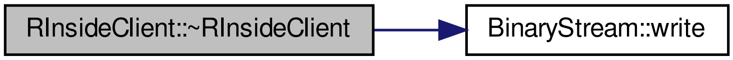
6.46.2.2 ∼RInsideClient()

```
RInsideClient::∼RInsideClient ( )
```

Definition at line 17 of file rinsideclient.cpp.

References can_send_command, had_unrecoverable_error, RIS_CMD_EXIT, stream, and BinaryStream::write().

Here is the call graph for this function:



6.46.3 Member Function Documentation

6.46.3.1 getConsoleOutput()

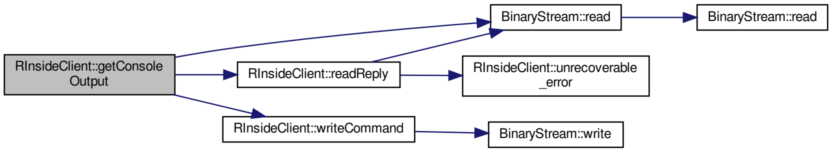
```
std::string RInsideClient::getConsoleOutput ( )
```

Definition at line 70 of file rinsideclient.cpp.

References can_send_command, BinaryStream::read(), readReply(), RIS_CMD_GETCONSOLE, stream, and write↩ Command().

Referenced by setCallback(), and test_console_output().

Here is the call graph for this function:


RInsideClient::getConsole Output
BinaryStream::read
BinaryStream::read
RInsideClient::readReply
RInsideClient::unrecoverable _error
RInsideClient::writeCommand
BinaryStream::write

6.46.3.2 getPlot()

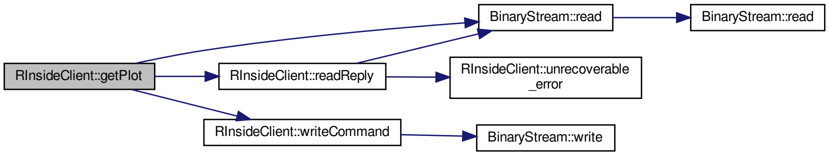
```
std::string RInsideClient::getPlot ( )
```

Definition at line 87 of file rinsideclient.cpp.

References can_send_command, BinaryStream::read(), readReply(), RIS_CMD_GETPLOT, stream, and write↩ Command().

Referenced by setCallback(), and test_plot().

Here is the call graph for this function:



6.46.3.3 getValue()

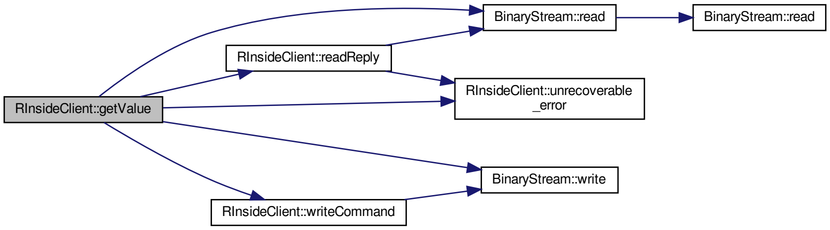
```
template<typename T >
T RInsideClient::getValue (
            const std::string & name )  [inline]
```

Definition at line 49 of file rinsideclient.h.

References can_send_command, BinaryStream::read(), readReply(), RIS_CMD_GETVALUE, unrecoverable_error(), BinaryStream::write(), and writeCommand().

Referenced by test_multiple(), and test_setting_getting().

Here is the call graph for this function:



6.46.3.4 initPlot()

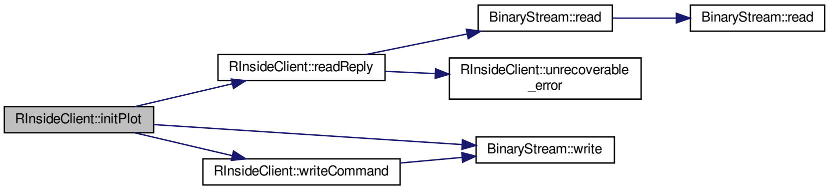
```
void RInsideClient::initPlot (
            uint32_t width = 800,
            uint32_t height = 600 )
```

Definition at line 79 of file rinsideclient.cpp.

References can_send_command, readReply(), RIS_CMD_INITPLOT, stream, BinaryStream::write(), and write←Command().

Referenced by setCallback(), and test_plot().

Here is the call graph for this function:



6.46.3.5 parseEval()

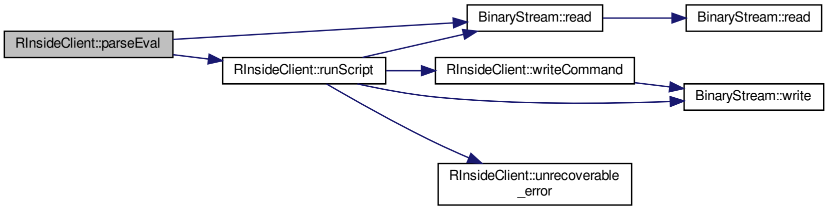
```
template<typename T >
T RInsideClient::parseEval (
             const std::string & code )  [inline]
```

Definition at line 30 of file rinsideclient.h.

References can_send_command, BinaryStream::read(), and runScript().

Referenced by test_callbacks(), and test_setting_getting().

Here is the call graph for this function:



6.46.3.6 parseEvalQ()

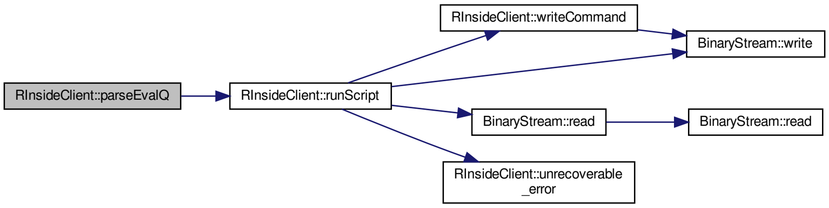
```
void RInsideClient::parseEvalQ (
             const std::string & code )  [inline]
```

Definition at line 25 of file rinsideclient.h.

References can_send_command, and runScript().

Referenced by test_console_output(), and test_plot().

Here is the call graph for this function:


RInsideClient::writeCommand
BinaryStream::write
RInsideClient::parseEvalQ
RInsideClient::runScript
BinaryStream::read
BinaryStream::read
RInsideClient::unrecoverable_error

6.46.3.7 readReply()

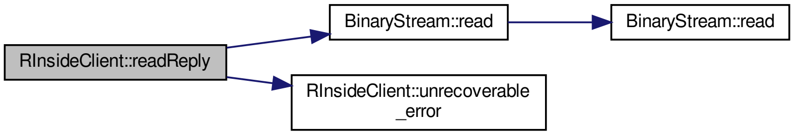
```
char RInsideClient::readReply (
             bool accept_ok = true,
             bool accept_value = false )  [private]
```

Definition at line 105 of file rinsideclient.cpp.

References can_send_command, BinaryStream::read(), RIS_REPLY_ERROR, RIS_REPLY_OK, RIS_REPLY_VAL↩ UE, stream, and unrecoverable_error().

Referenced by getConsoleOutput(), getPlot(), getValue(), initPlot(), setCallback(), and setValue().

Here is the call graph for this function:



6.46.3.8 runScript()

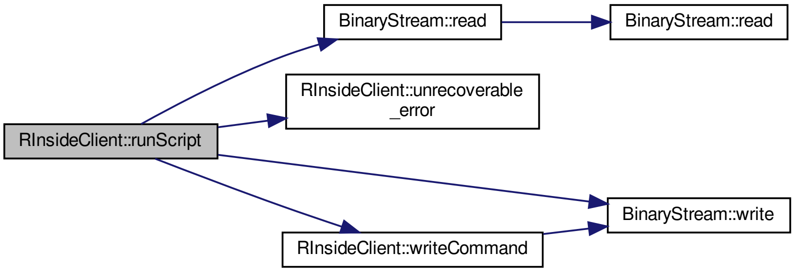
```
void RInsideClient::runScript (
             const std::string code,
             int32_t result_typeid )  [private]
```

Definition at line 29 of file rinsideclient.cpp.

References callbacks, can_send_command, had_unrecoverable_error, BinaryStream::read(), RIS_CMD_RUN, RIS↩ _REPLY_CALLBACK, RIS_REPLY_OK, RIS_REPLY_VALUE, stream, unrecoverable_error(), BinaryStream::write(), and writeCommand().

Referenced by parseEval(), parseEvalQ(), and setCallback().

Here is the call graph for this function:



6.46.3.9 setCallback()

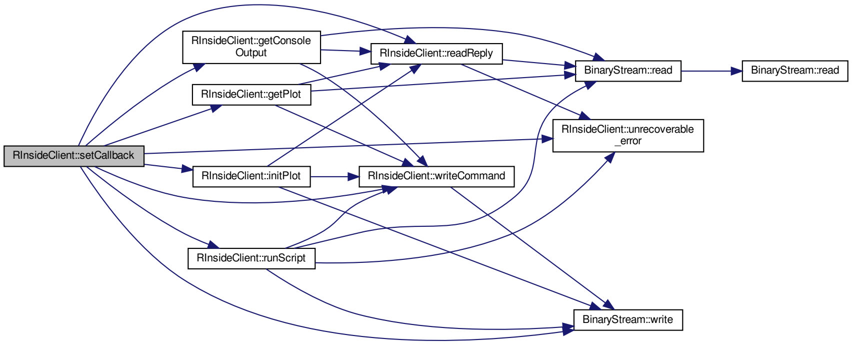
```
template<typename R , typename...  Params>
void RInsideClient::setCallback (
            const std::string & name,
            std::function< R(Params...)> & callback )  [inline]
```

Definition at line 65 of file rinsideclient.h.

References callbacks, can_send_command, getConsoleOutput(), getPlot(), initPlot(), next_callback_id, readReply(), RIS_CMD_SETCALLBACK, runScript(), unrecoverable_error(), BinaryStream::write(), and writeCommand().

Referenced by test_callbacks().

Here is the call graph for this function:



6.46.3.10 setValue()

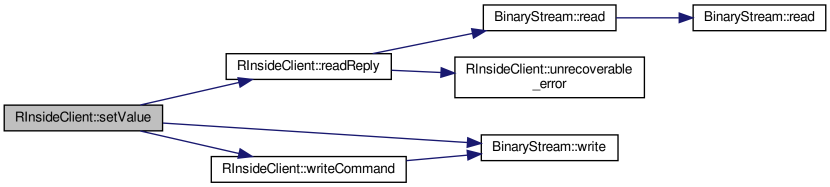
```
template<typename T >
void RInsideClient::setValue (
            const std::string & name,
            const T & value )  [inline]
```

Definition at line 38 of file rinsideclient.h.

References can_send_command, readReply(), RIS_CMD_SETVALUE, BinaryStream::write(), and writeCommand().

Referenced by test_multiple(), and test_setting_getting().

Here is the call graph for this function:



6.46.3.11 unrecoverable_error()

```
void RInsideClient::unrecoverable_error (
             const std::string & error )  [private]
```

Definition at line 120 of file rinsideclient.cpp.

References had_unrecoverable_error.

Referenced by getValue(), readReply(), runScript(), and setCallback().

6.46.3.12 writeCommand()

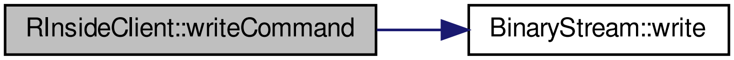
```
void RInsideClient::writeCommand (
             char command )  [private]
```

Definition at line 95 of file rinsideclient.cpp.

References can_send_command, had_unrecoverable_error, stream, and BinaryStream::write().

Referenced by getConsoleOutput(), getPlot(), getValue(), initPlot(), runScript(), setCallback(), and setValue().

Here is the call graph for this function:



6.46.4 Member Data Documentation

6.46.4.1 callbacks

```
std::map<uint32_t, std::function<void(void)> > RInsideClient::callbacks  [private]
```

Definition at line 98 of file rinsideclient.h.

Referenced by runScript(), and setCallback().

6.46.4.2 can_send_command

```
bool RInsideClient::can_send_command  [private]
```

Definition at line 100 of file rinsideclient.h.

Referenced by getConsoleOutput(), getPlot(), getValue(), initPlot(), parseEval(), parseEvalQ(), readReply(), RInside↩ Client(), runScript(), setCallback(), setValue(), writeCommand(), and ∼RInsideClient().

6.46.4.3 had_unrecoverable_error

```
bool RInsideClient::had_unrecoverable_error  [private]
```

Definition at line 99 of file rinsideclient.h.

Referenced by runScript(), unrecoverable_error(), writeCommand(), and ∼RInsideClient().

6.46.4.4 next_callback_id

```
uint32_t RInsideClient::next_callback_id  [private]
```

Definition at line 97 of file rinsideclient.h.

Referenced by setCallback().

6.46.4.5 stream

```
BinaryStream RInsideClient::stream  [private]
```

Definition at line 96 of file rinsideclient.h.

Referenced by getConsoleOutput(), getPlot(), initPlot(), readReply(), RInsideClient(), runScript(), writeCommand(), and ∼RInsideClient().

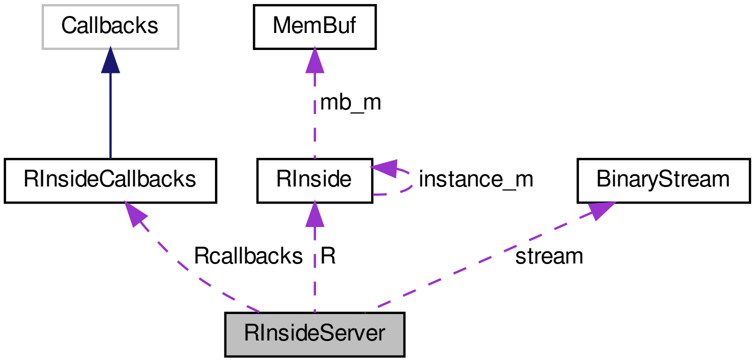
The documentation for this class was generated from the following files:

- inst/examples/sandboxed_server/client/rinsideclient.h
- inst/examples/sandboxed_server/client/rinsideclient.cpp

6.47 RInsideServer Class Reference

```
#include <rinsideserver.h>
```

Collaboration diagram for RInsideServer:



Public Member Functions

- RInsideServer (BinaryStream &stream, RInside &R, RInsideCallbacks &Rcallbacks)
- ∼RInsideServer ()
- void run ()

Static Public Member Functions

- static void registerDefaultTypes ()
- template<typename T >
  static void registerType ()

Private Member Functions

- SEXP sexp_from_stream ()
- void sexp_to_stream (SEXP, int32_t type, bool include_reply=false)
- void sendReply (char reply)
- void allowSendReply ()

Private Attributes

- BinaryStream stream
- RInside & R
- RInsideCallbacks & Rcallbacks
- bool can_send_reply

Static Private Attributes

- static std::map< int32_t, std::function< SEXP(BinaryStream &)> > registry_sexp_from_stream
- static std::map< int32_t, std::function< void(RInsideServer &, SEXP, bool)> > registry_sexp_to_stream

Friends

- class CppFunctionForRInsideServer

6.47.1 Detailed Description

Definition at line 19 of file rinsideserver.h.

6.47.2 Constructor & Destructor Documentation

6.47.2.1 RInsideServer()

```
RInsideServer::RInsideServer (
                BinaryStream & stream,
                RInside & R,
                RInsideCallbacks & Rcallbacks )
```

Definition at line 45 of file rinsideserver.cpp.

6.47.2.2 ∼RInsideServer()

```
RInsideServer::∼RInsideServer ( )
```

Definition at line 49 of file rinsideserver.cpp.

6.47.3 Member Function Documentation

6.47.3.1 allowSendReply()

```
void RInsideServer::allowSendReply ( )  [inline], [private]
```

Definition at line 36 of file rinsideserver.h.

Referenced by CppFunctionForRInsideServer::operator()(), and run().

6.47.3.2 registerDefaultTypes()

```
void RInsideServer::registerDefaultTypes ( )  [static]
```

Definition at line 202 of file rinsideserver.cpp.

Referenced by main().

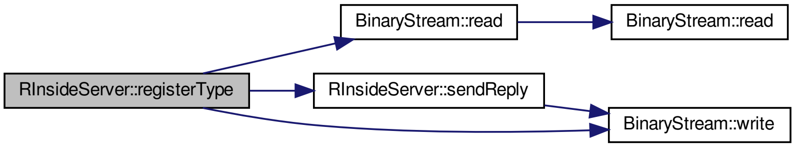
6.47.3.3 registerType()

```
template<typename T >
static void RInsideServer::registerType ( )  [inline], [static]
```

Definition at line 44 of file rinsideserver.h.

References BinaryStream::read(), RIS_REPLY_VALUE, sendReply(), stream, and BinaryStream::write().

Here is the call graph for this function:



6.47.3.4 run()

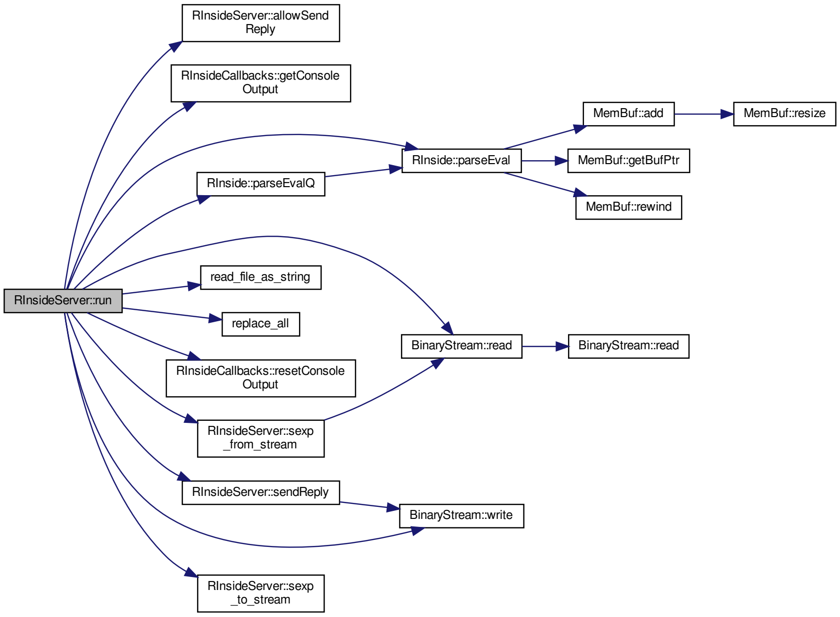
```
void RInsideServer::run ( )
```

Definition at line 63 of file rinsideserver.cpp.

References allowSendReply(), CMD_CATCH, CMD_TRY, RInsideCallbacks::getConsoleOutput(), LOG, RInside↩ ::parseEval(), RInside::parseEvalQ(), R, Rcallbacks, BinaryStream::read(), read_file_as_string(), replace_all(), R↩ InsideCallbacks::resetConsoleOutput(), RIS_CMD_EXIT, RIS_CMD_GETCONSOLE, RIS_CMD_GETPLOT, RIS_C↩ MD_GETVALUE, RIS_CMD_INITPLOT, RIS_CMD_RUN, RIS_CMD_SETCALLBACK, RIS_CMD_SETVALUE, RI↩ S_MAGIC_NUMBER, RIS_REPLY_OK, RIS_REPLY_VALUE, sendReply(), sexp_from_stream(), sexp_to_stream(), stream, and BinaryStream::write().

Referenced by main().

Here is the call graph for this function:



6.47.3.5 sendReply()

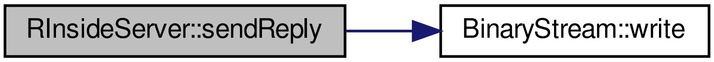
```
void RInsideServer::sendReply (
            char reply )  [inline], [private]
```

Definition at line 35 of file rinsideserver.h.

References BinaryStream::write().

Referenced by CppFunctionForRInsideServer::operator()(), registerType(), and run().

Here is the call graph for this function:



6.47.3.6 sexp_from_stream()

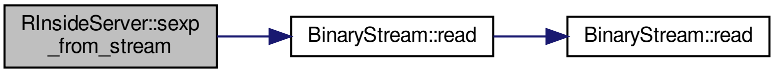
```
SEXP RInsideServer::sexp_from_stream ( )  [private]
```

Definition at line 182 of file rinsideserver.cpp.

References LOG, BinaryStream::read(), registry_sexp_from_stream, and stream.

Referenced by CppFunctionForRInsideServer::operator()(), and run().

Here is the call graph for this function:



6.47.3.7 sexp_to_stream()

```
void RInsideServer::sexp_to_stream (
             SEXP sexp,
             int32_t type,
             bool include_reply = false )  [private]
```

Definition at line 193 of file rinsideserver.cpp.

References LOG, and registry_sexp_to_stream.

Referenced by CppFunctionForRInsideServer::operator()(), and run().

6.47.4 Friends And Related Function Documentation

6.47.4.1 CppFunctionForRInsideServer

```
friend class CppFunctionForRInsideServer  [friend]
```

Definition at line 68 of file rinsideserver.h.

6.47.5 Member Data Documentation

6.47.5.1 can_send_reply

```
bool RInsideServer::can_send_reply  [private]
```

Definition at line 34 of file rinsideserver.h.

6.47.5.2 R

```
RInside& RInsideServer::R  [private]
```

Definition at line 31 of file rinsideserver.h.

Referenced by run().

6.47.5.3 Rcallbacks

```
RInsideCallbacks& RInsideServer::Rcallbacks  [private]
```

Definition at line 32 of file rinsideserver.h.

Referenced by run().

6.47.5.4 registry_sexp_from_stream

```
std::map< int32_t, std::function< SEXP(BinaryStream &)> > RInsideServer::registry_sexp_from_↩
stream  [static], [private]
```

Definition at line 38 of file rinsideserver.h.

Referenced by read_file_as_string(), and sexp_from_stream().

6.47.5.5 registry_sexp_to_stream

```
std::map< int32_t, std::function< void(RInsideServer &, SEXP, bool)> > RInsideServer::registry↩
_sexp_to_stream  [static], [private]
```

Definition at line 39 of file rinsideserver.h.

Referenced by read_file_as_string(), and sexp_to_stream().

6.47.5.6 stream

```
BinaryStream RInsideServer::stream  [private]
```

Definition at line 30 of file rinsideserver.h.

Referenced by CppFunctionForRInsideServer::operator()(), registerType(), run(), and sexp_from_stream().

The documentation for this class was generated from the following files:

- inst/examples/sandboxed_server/server/rinsideserver.h
- inst/examples/sandboxed_server/server/rinsideserver.cpp

6.48 callback_helper::send_pack< Params > Struct Template Reference

```
#include <callback_helper.h>
```

6.48.1 Detailed Description

template<typename... Params>
struct callback_helper::send_pack< Params >

Definition at line 9 of file callback_helper.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/client/callback_helper.h

6.49 callback_helper::send_pack< First, Remaining... > Struct Template Reference

```
#include <callback_helper.h>
```

Static Public Member Functions

- static void send (BinaryStream &stream)

6.49.1 Detailed Description

template<typename First, typename... Remaining>
struct callback_helper::send_pack< First, Remaining... >

Definition at line 12 of file callback_helper.h.

6.49.2 Member Function Documentation

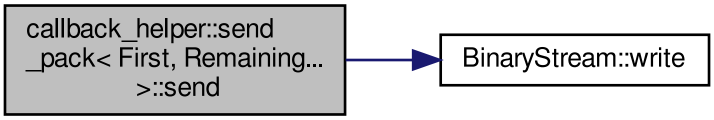
6.49.2.1 send()

```
template<typename First , typename...  Remaining>
static void callback_helper::send_pack< First, Remaining...  >::send (
             BinaryStream & stream )  [inline], [static]
```

Definition at line 13 of file callback_helper.h.

References BinaryStream::write().

Here is the call graph for this function:



The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/client/callback_helper.h

6.50 callback_helper::send_pack<> Struct Template Reference

```
#include <callback_helper.h>
```

Static Public Member Functions

- static void send (BinaryStream &stream)

6.50.1 Detailed Description

template<>
struct callback_helper::send_pack<>

Definition at line 21 of file callback_helper.h.

6.50.2 Member Function Documentation

6.50.2.1 send()

```
static void callback_helper::send_pack<>::send (
              BinaryStream & stream )  [inline], [static]
```

Definition at line 22 of file callback_helper.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/client/callback_helper.h

6.51 serialization::serializer< T > Struct Template Reference

```
#include <binarystream.h>
```

6.51.1 Detailed Description

template<typename T>
struct serialization::serializer< T >

Definition at line 81 of file binarystream.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/binarystream.h

6.52 serialization::serializer< std::string > Struct Template Reference

```
#include <binarystream.h>
```

Static Public Member Functions

- static void serialize (BinaryStream &, const std::string &)
- static std::string deserialize (BinaryStream &)

6.52.1 Detailed Description

template<>
struct serialization::serializer< std::string >

Definition at line 84 of file binarystream.h.

6.52.2 Member Function Documentation

6.52.2.1 deserialize()

```
std::string serialization::serializer< std::string >::deserialize (
             BinaryStream & stream )  [static]
```

Definition at line 124 of file binarystream.cpp.

References BinaryStream::read().

Here is the call graph for this function:

serialization::serializer
< std::string >::deserialize
BinaryStream::read
BinaryStream::read

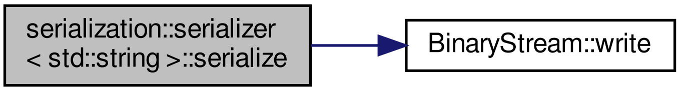
6.52.2.2 serialize()

```
void serialization::serializer< std::string >::serialize (
            BinaryStream & stream,
            const std::string & string )  [static]
```

Definition at line 116 of file binarystream.cpp.

References BinaryStream::write().

Here is the call graph for this function:



The documentation for this struct was generated from the following files:

- inst/examples/sandboxed_server/common/binarystream.h
- inst/examples/sandboxed_server/common/binarystream.cpp

6.53 serialization::serializer< std::vector< T > > Struct Template Reference

```
#include <binarystream.h>
```

Static Public Member Functions

- static void serialize (BinaryStream &, const std::vector< T > &)
- static std::vector< T > deserialize (BinaryStream &)

6.53.1 Detailed Description

template<typename T>
struct serialization::serializer< std::vector< T > >

Definition at line 90 of file binarystream.h.

6.53.2 Member Function Documentation

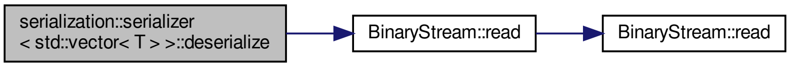
6.53.2.1 deserialize()

```
template<typename T >
std::vector< T > serialization::serializer< std::vector< T > >::deserialize (
             BinaryStream & stream )  [static]
```

Definition at line 150 of file binarystream.cpp.

References BinaryStream::read().

Here is the call graph for this function:



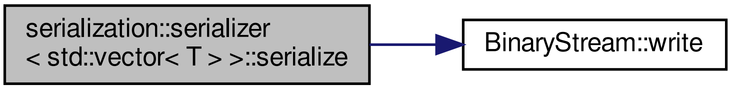
6.53.2.2 serialize()

```
template<typename T >
void serialization::serializer< std::vector< T > >::serialize (
             BinaryStream & stream,
             const std::vector< T > & vec )  [static]
```

Definition at line 141 of file binarystream.cpp.

References BinaryStream::write().

Here is the call graph for this function:



The documentation for this struct was generated from the following files:

- inst/examples/sandboxed_server/common/binarystream.h
- inst/examples/sandboxed_server/common/binarystream.cpp

6.54 Solver Class Reference

Public Member Functions

- Solver (int n)
- void Iterate (int n)

Private Types

- typedef std::vector< Planet > Planets

Private Member Functions

- void Iteration ()

Private Attributes

- Planets tab
- double dt
- double G

Friends

- class Wrapper

6.54.1 Detailed Description

Definition at line 23 of file rinside_interactive0.cpp.

6.54.2 Member Typedef Documentation

6.54.2.1 Planets

```
typedef std::vector<Planet> Solver::Planets  [private]
```

Definition at line 24 of file rinside_interactive0.cpp.

6.54.3 Constructor & Destructor Documentation

6.54.3.1 Solver()

```
Solver::Solver (
            int n )  [inline]
```

Definition at line 49 of file rinside_interactive0.cpp.

6.54.4 Member Function Documentation

6.54.4.1 Iterate()

```
void Solver::Iterate (
            int n )  [inline]
```

Definition at line 58 of file rinside_interactive0.cpp.

Referenced by main().

6.54.4.2 Iteration()

```
void Solver::Iteration ( )  [inline], [private]
```

Definition at line 28 of file rinside_interactive0.cpp.

References Planet::x, and Planet::y.

6.54.5 Friends And Related Function Documentation

6.54.5.1 Wrapper

```
friend class Wrapper  [friend]
```

Definition at line 61 of file rinside_interactive0.cpp.

6.54.6 Member Data Documentation

6.54.6.1 dt

```
double Solver::dt  [private]
```

Definition at line 26 of file rinside_interactive0.cpp.

Referenced by Wrapper::dt().

6.54.6.2 G

```
double Solver::G  [private]
```

Definition at line 27 of file rinside_interactive0.cpp.

Referenced by Wrapper::G().

6.54.6.3 tab

```
Planets Solver::tab  [private]
```

Definition at line 25 of file rinside_interactive0.cpp.

Referenced by Wrapper::getData(), and Wrapper::setData().

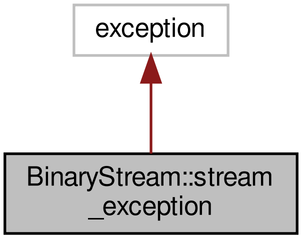
The documentation for this class was generated from the following file:

- inst/examples/standard/rinside_interactive0.cpp

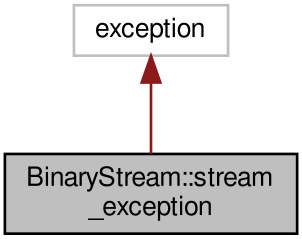
6.55 BinaryStream::stream_exception Class Reference

```
#include <binarystream.h>
```

Inheritance diagram for BinaryStream::stream_exception:



Collaboration diagram for BinaryStream::stream_exception:



6.55.1 Detailed Description

Definition at line 68 of file binarystream.h.

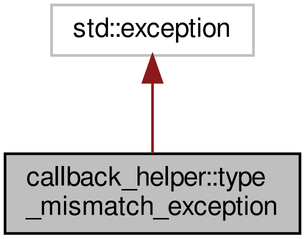
The documentation for this class was generated from the following file:

- inst/examples/sandboxed_server/common/binarystream.h

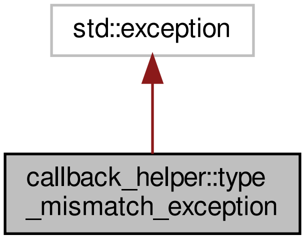
6.56 callback_helper::type_mismatch_exception Class Reference

```
#include <callback_helper.h>
```

Inheritance diagram for callback_helper::type_mismatch_exception:



Collaboration diagram for callback_helper::type_mismatch_exception:



6.56.1 Detailed Description

Definition at line 27 of file callback_helper.h.

The documentation for this class was generated from the following file:

- inst/examples/sandboxed_server/client/callback_helper.h

6.57 typeid_helpers::void_t_struct< ... > Struct Template Reference

```
#include <typeid.h>
```

Public Types

- using type = void

6.57.1 Detailed Description

template<typename...>
struct typeid_helpers::void_t_struct< ... >

Definition at line 26 of file typeid.h.

6.57.2 Member Typedef Documentation

6.57.2.1 type

```
template<typename... >
using typeid_helpers::void_t_struct<... >::type = void
```

Definition at line 26 of file typeid.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/typeid.h

6.58 binary_stream_helpers::void_t_struct< ... > Struct Template Reference

```
#include <binarystream.h>
```

Public Types

- using type = void

6.58.1 Detailed Description

template<typename...>
struct binary_stream_helpers::void_t_struct<... >

Definition at line 105 of file binarystream.h.

6.58.2 Member Typedef Documentation

6.58.2.1 type

```
template<typename...  >
using binary_stream_helpers::void_t_struct<...  >::type = void
```

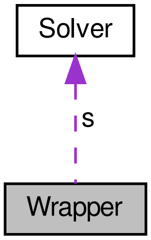
Definition at line 105 of file binarystream.h.

The documentation for this struct was generated from the following file:

- inst/examples/sandboxed_server/common/binarystream.h

6.59 Wrapper Class Reference

Collaboration diagram for Wrapper:



Public Member Functions

- Wrapper (Solver ∗s_)
- Rcpp::DataFrame getData ()
- void setData (Rcpp::DataFrame tab)
- double & G ()
- double & dt ()

Private Attributes

- Solver ∗ s

6.59.1 Detailed Description

Definition at line 65 of file rinside_interactive0.cpp.

6.59.2 Constructor & Destructor Documentation

6.59.2.1 Wrapper()

```
Wrapper::Wrapper (
             Solver * s_ )  [inline]
```

Definition at line 68 of file rinside_interactive0.cpp.

6.59.3 Member Function Documentation

6.59.3.1 dt()

```
double& Wrapper::dt ( )  [inline]
```

Definition at line 104 of file rinside_interactive0.cpp.

References Solver::dt.

6.59.3.2 G()

```
double& Wrapper::G ( )  [inline]
```

Definition at line 101 of file rinside_interactive0.cpp.

References Solver::G.

6.59.3.3 getData()

```
Rcpp::DataFrame Wrapper::getData ( )  [inline]
```

Definition at line 69 of file rinside_interactive0.cpp.

References Planet::m, Solver::tab, Planet::vx, Planet::vy, Planet::x, and Planet::y.

6.59.3.4 setData()

```
void Wrapper::setData (
            Rcpp::DataFrame tab )  [inline]
```

Definition at line 84 of file rinside_interactive0.cpp.

References Planet::m, Solver::tab, Planet::vx, Planet::vy, Planet::x, and Planet::y.

6.59.4 Member Data Documentation

6.59.4.1 s

```
Solver* Wrapper::s  [private]
```

Definition at line 66 of file rinside_interactive0.cpp.

The documentation for this class was generated from the following file:

- inst/examples/standard/rinside_interactive0.cpp

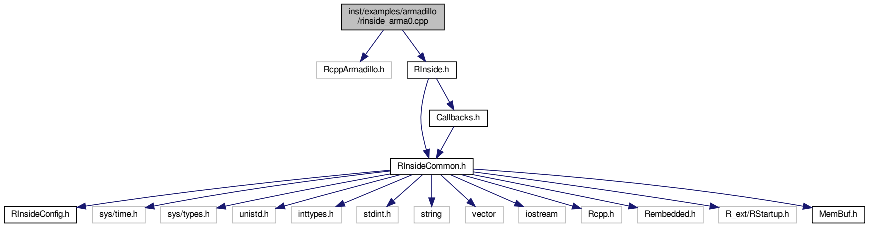
Chapter 7

File Documentation

7.1 inst/examples/armadillo/rinside_arma0.cpp File Reference

```
#include <RcppArmadillo.h>
#include <RInside.h>
```

Include dependency graph for rinside_arma0.cpp:



Functions

- int main (int argc, char $*$argv[ ])

7.1.1 Function Documentation

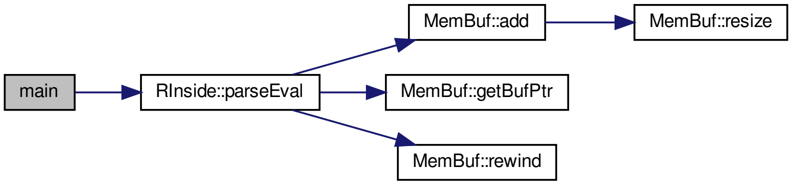
7.1.1.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Definition at line 10 of file rinside_arma0.cpp.

References RInside::parseEval().

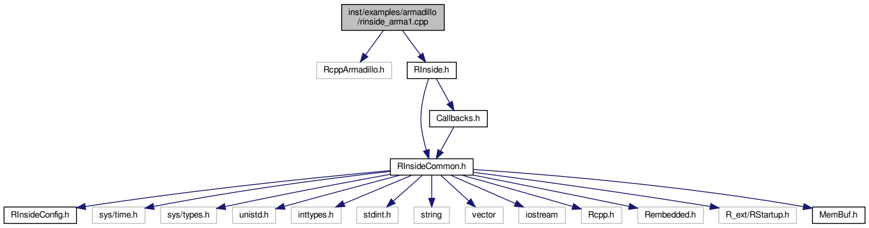
Here is the call graph for this function:



7.2 inst/examples/armadillo/rinside_arma1.cpp File Reference

```
#include <RcppArmadillo.h>
#include <RInside.h>
```

Include dependency graph for rinside_arma1.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.2.1 Function Documentation

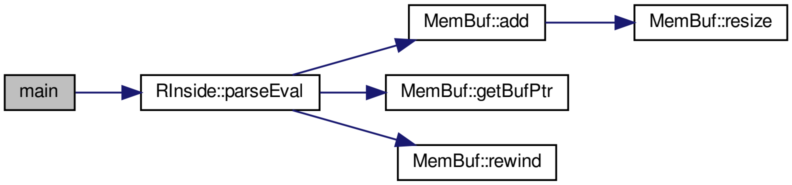
7.2.1.1 main()

```
int main (
             int argc,
             char * argv[] )
```

Definition at line 10 of file rinside_arma1.cpp.

References RInside::parseEval().

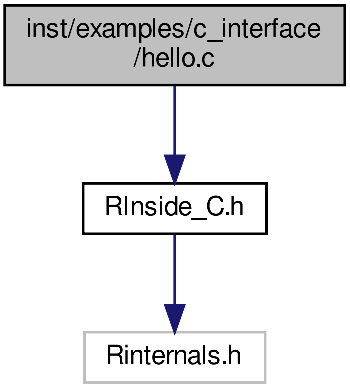
Here is the call graph for this function:



7.3 inst/examples/c_interface/hello.c File Reference

```
#include <RInside_C.h>
```

Include dependency graph for hello.c:



Functions

- int main ()

7.3.1 Function Documentation

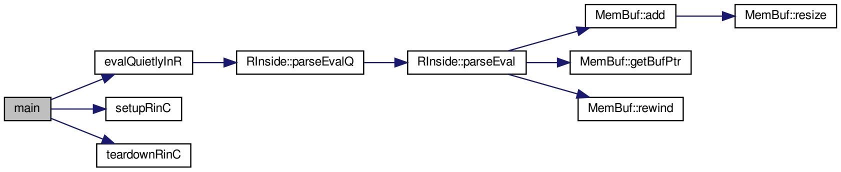
7.3.1.1 main()

```
int main (
            void  )
```

Definition at line 3 of file hello.c.

References evalQuietlyInR(), setupRinC(), and teardownRinC().

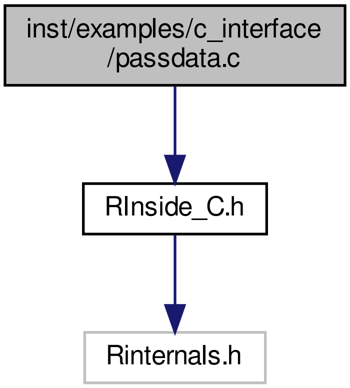
Here is the call graph for this function:



7.4 inst/examples/c_interface/passdata.c File Reference

```
#include <RInside_C.h>
```

Include dependency graph for passdata.c:



Functions

- int main ()

7.4.1 Function Documentation

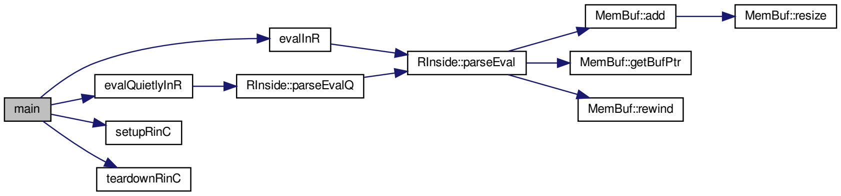
7.4.1.1 main()

```
int main (
             void  )
```

Definition at line 3 of file passdata.c.

References evalInR(), evalQuietlyInR(), setupRinC(), and teardownRinC().

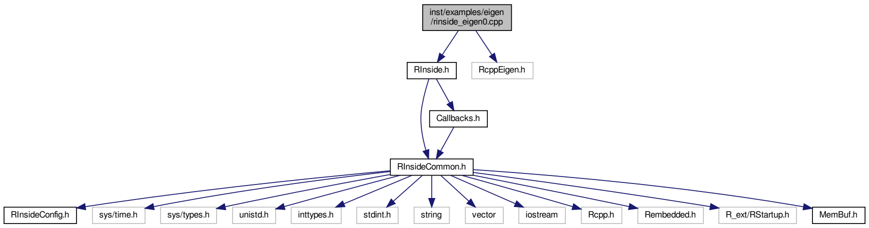
Here is the call graph for this function:



7.5 inst/examples/eigen/rinside_eigen0.cpp File Reference

```
#include <RInside.h>
#include <RcppEigen.h>
```

Include dependency graph for rinside_eigen0.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.5.1 Function Documentation

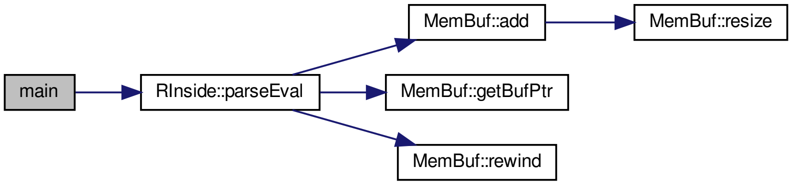
7.5.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 10 of file rinside_eigen0.cpp.

References RInside::parseEval().

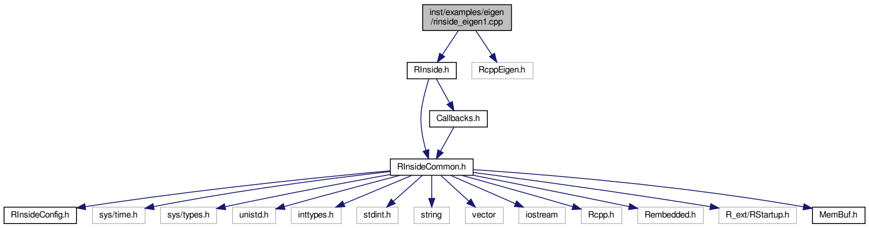
Here is the call graph for this function:



7.6 inst/examples/eigen/rinside_eigen1.cpp File Reference

```
#include <RInside.h>
#include <RcppEigen.h>
```

Include dependency graph for rinside_eigen1.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.6.1 Function Documentation

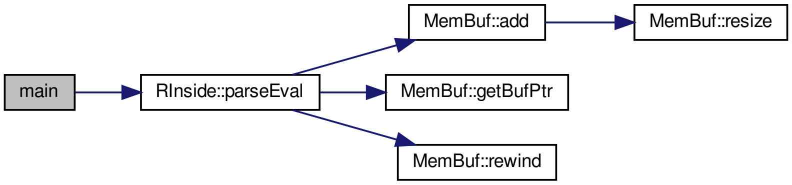
7.6.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 10 of file rinside_eigen1.cpp.

References RInside::parseEval().

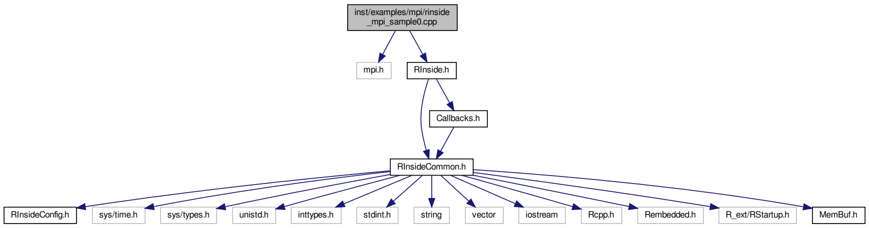
Here is the call graph for this function:



7.7 inst/examples/mpi/rinside_mpi_sample0.cpp File Reference

```
#include <mpi.h>
#include <RInside.h>
```

Include dependency graph for rinside_mpi_sample0.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.7.1 Function Documentation

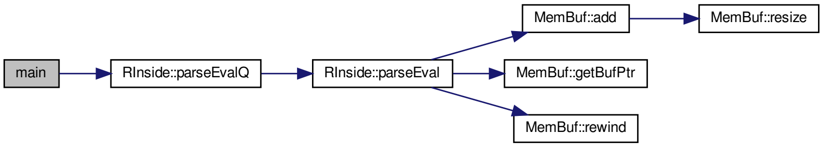
7.7.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 14 of file rinside_mpi_sample0.cpp.

References RInside::parseEvalQ().

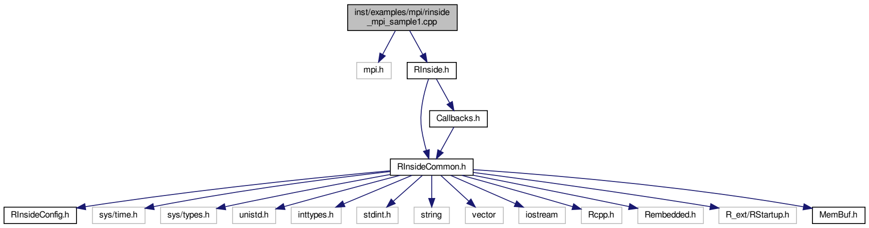
Here is the call graph for this function:



7.8 inst/examples/mpi/rinside_mpi_sample1.cpp File Reference

```
#include <mpi.h>
#include <RInside.h>
```

Include dependency graph for rinside_mpi_sample1.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.8.1 Function Documentation

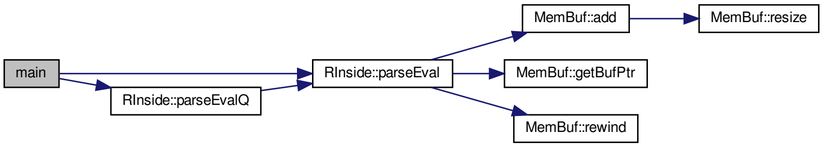
7.8.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 14 of file rinside_mpi_sample1.cpp.

References RInside::parseEval(), and RInside::parseEvalQ().

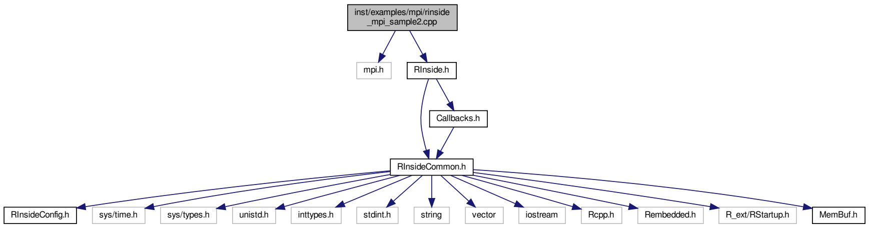
Here is the call graph for this function:



7.9 inst/examples/mpi/rinside_mpi_sample2.cpp File Reference

```
#include <mpi.h>
#include <RInside.h>
```

Include dependency graph for rinside_mpi_sample2.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.9.1 Function Documentation

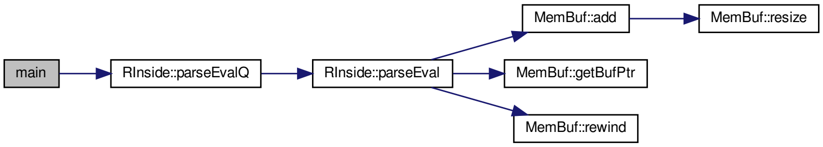
7.9.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 14 of file rinside_mpi_sample2.cpp.

References RInside::parseEvalQ().

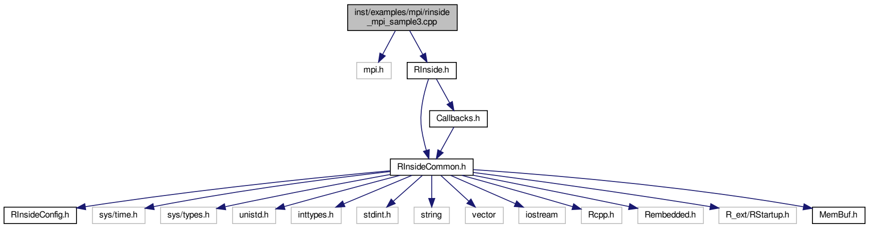
Here is the call graph for this function:



7.10 inst/examples/mpi/rinside_mpi_sample3.cpp File Reference

```
#include <mpi.h>
#include <RInside.h>
```

Include dependency graph for rinside_mpi_sample3.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.10.1 Function Documentation

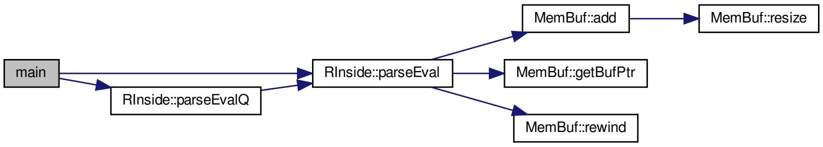
7.10.1.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 14 of file rinside_mpi_sample3.cpp.

References RInside::parseEval(), and RInside::parseEvalQ().

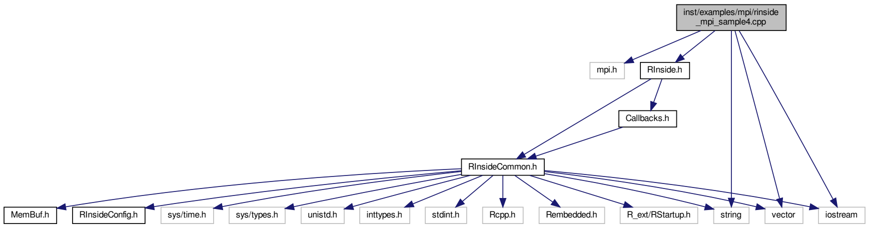
Here is the call graph for this function:



7.11 inst/examples/mpi/rinside_mpi_sample4.cpp File Reference

```
#include <mpi.h>
#include <RInside.h>
#include <string>
#include <vector>
#include <iostream>
```

Include dependency graph for rinside_mpi_sample4.cpp:



Macros

- #define WORKTAG 1
- #define DIETAG 2

Functions

- static void master (void)
- static void slave (RInside &R)
- static int get_next_work_item (int &work, const int size_work, std::vector< int > &data)
- static void do_work (int work, int &result, RInside &R)
- static void initalize (RInside &R)
- int main (int argc, char ∗∗argv)

Variables

- int itr = 0

7.11.1 Macro Definition Documentation

7.11.1.1 DIETAG

```
#define DIETAG 2
```

Definition at line 20 of file rinside_mpi_sample4.cpp.

Referenced by master(), and slave().

7.11.1.2 WORKTAG

```
#define WORKTAG 1
```

Definition at line 19 of file rinside_mpi_sample4.cpp.

Referenced by master().

7.11.2 Function Documentation

7.11.2.1 do_work()

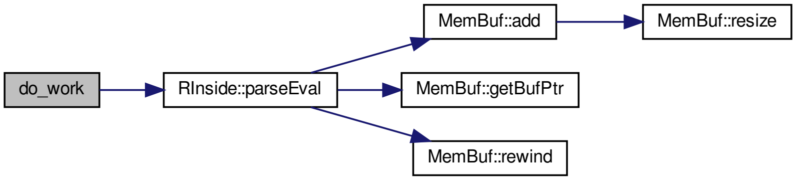
```
static void do_work (
             int work,
             int & result,
             RInside & R )  [static]
```

Definition at line 133 of file rinside_mpi_sample4.cpp.

References RInside::parseEval().

Referenced by slave().

Here is the call graph for this function:



7.11.2.2 get_next_work_item()

```
static int get_next_work_item (
             int & work,
             const int size_work,
             std::vector< int > & data )  [static]
```

Definition at line 120 of file rinside_mpi_sample4.cpp.

References itr.

Referenced by master().

7.11.2.3 initalize()

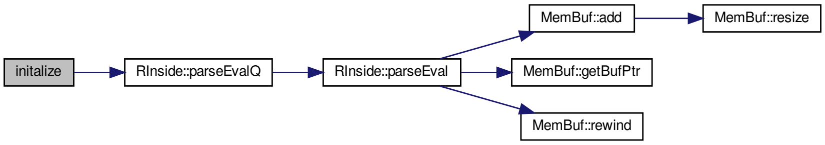
```
static void initalize (
            RInside & R )  [static]
```

Definition at line 49 of file rinside_mpi_sample4.cpp.

References RInside::parseEvalQ().

Referenced by main().

Here is the call graph for this function:



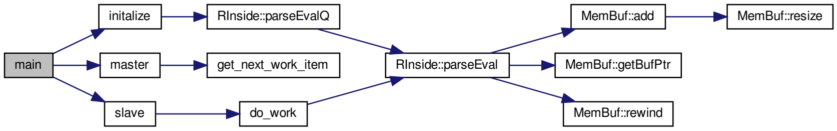
7.11.2.4 main()

```
int main (
            int argc,
            char ** argv )
```

Definition at line 31 of file rinside_mpi_sample4.cpp.

References initalize(), master(), and slave().

Here is the call graph for this function:



7.11.2.5 master()

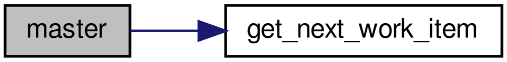
```
static void master (
            void  )  [static]
```

Definition at line 55 of file rinside_mpi_sample4.cpp.

References DIETAG, get_next_work_item(), and WORKTAG.

Referenced by main().

Here is the call graph for this function:



7.11.2.6 slave()

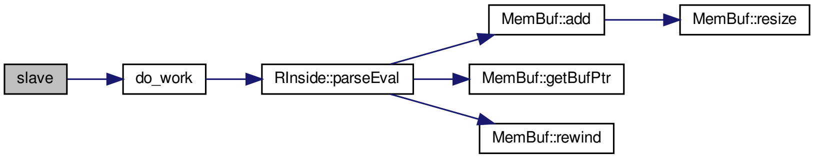
```
static void slave (
            RInside & R )  [static]
```

Definition at line 99 of file rinside_mpi_sample4.cpp.

References DIETAG, and do_work().

Referenced by main().

Here is the call graph for this function:



7.11.3 Variable Documentation

7.11.3.1 itr

```
int itr = 0
```

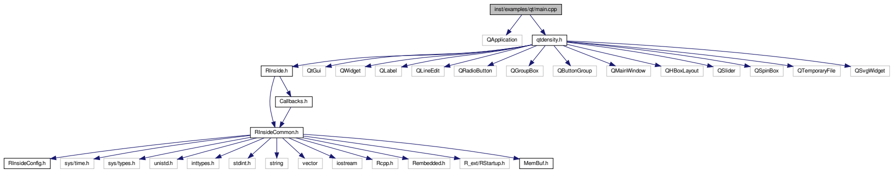
Definition at line 29 of file rinside_mpi_sample4.cpp.

Referenced by get_next_work_item().

7.12 inst/examples/qt/main.cpp File Reference

```
#include <QApplication>
#include "qtdensity.h"
```

Include dependency graph for main.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.12.1 Function Documentation

7.12.1.1 main()

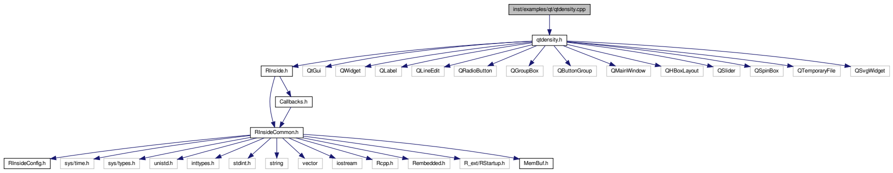
```
int main (
             int argc,
             char ∗ argv[ ] )
```

Definition at line 12 of file main.cpp.

7.13 inst/examples/qt/qtdensity.cpp File Reference

```
#include "qtdensity.h"
```

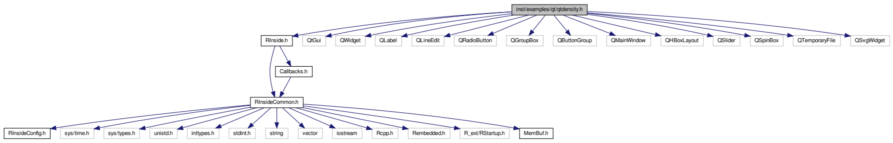
Include dependency graph for qtdensity.cpp:



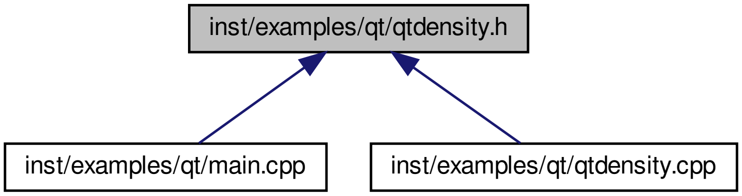
7.14 inst/examples/qt/qtdensity.h File Reference

```
#include <RInside.h>
#include <QtGui>
#include <QWidget>
#include <QLabel>
#include <QLineEdit>
#include <QRadioButton>
#include <QGroupBox>
#include <QButtonGroup>
#include <QMainWindow>
#include <QHBoxLayout>
#include <QSlider>
#include <QSpinBox>
#include <QTemporaryFile>
#include <QSvgWidget>
```

Include dependency graph for qtdensity.h:



This graph shows which files directly or indirectly include this file:

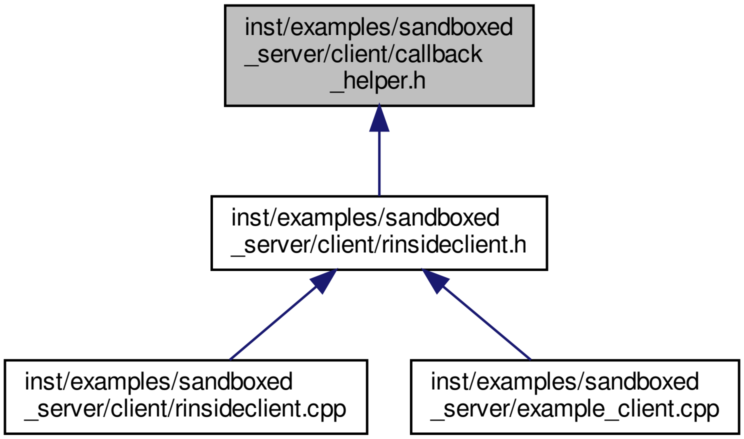

inst/examples/qt/qtdensity.h
inst/examples/qt/main.cpp
inst/examples/qt/qtdensity.cpp

Classes

- class QtDensity

7.15 inst/examples/sandboxed_server/client/callback_helper.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- struct callback_helper::send_pack< Params >
- struct callback_helper::send_pack< First, Remaining... >
- struct callback_helper::send_pack<>
- class callback_helper::type_mismatch_exception
- class callback_helper::parameter_error_exception

Namespaces

- callback_helper

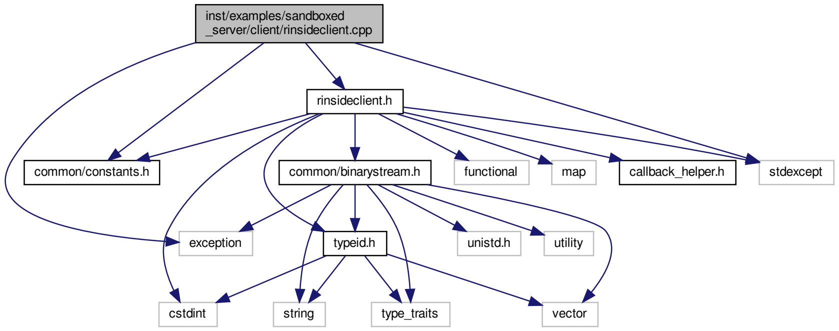
Functions

- template<typename T >
  T callback_helper::read_from_stream_with_typeid (BinaryStream &stream)
- template<typename RESULT_TYPE >
  void callback_helper::call (const std::function< RESULT_TYPE()> &fun, BinaryStream &stream)
- template<typename RESULT_TYPE , typename U0 >
  void callback_helper::call (const std::function< RESULT_TYPE(U0)> &fun, BinaryStream &stream)
- template<typename RESULT_TYPE , typename U0 , typename U1 >
  void callback_helper::call (const std::function< RESULT_TYPE(U0, U1)> &fun, BinaryStream &stream)
- template<typename RESULT_TYPE , typename U0 , typename U1 , typename U2 >
  void callback_helper::call (const std::function< RESULT_TYPE(U0, U1, U2)> &fun, BinaryStream &stream)

7.16 inst/examples/sandboxed_server/client/rinsideclient.cpp File Reference

```
#include "rinsideclient.h"
#include "common/constants.h"
#include <exception>
#include <stdexcept>
```

Include dependency graph for rinsideclient.cpp:

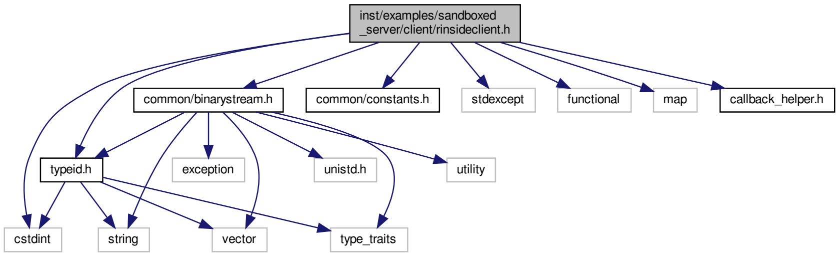


7.17 inst/examples/sandboxed_server/client/rinsideclient.h File Reference

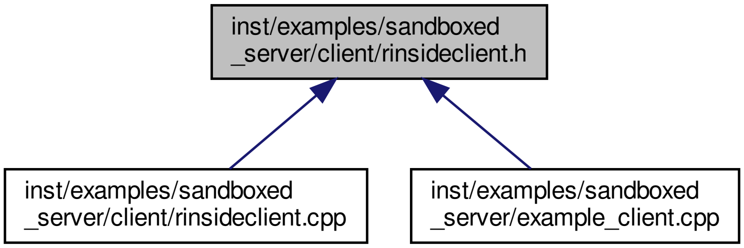
```
#include "common/binarystream.h"
#include "common/constants.h"
#include "common/typeid.h"
#include <stdexcept>
#include <cstdint>
```

```
#include <functional>
#include <map>
#include "callback_helper.h"
```

Include dependency graph for rinsideclient.h:



This graph shows which files directly or indirectly include this file:


inst/examples/sandboxed
_server/client/rinsideclient.h
inst/examples/sandboxed
_server/client/rinsideclient.cpp
inst/examples/sandboxed
_server/example_client.cpp

Classes

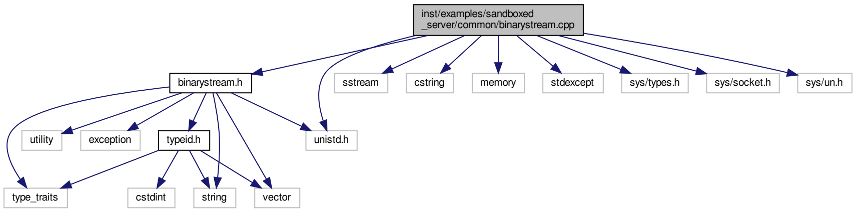
- class RInsideClient

7.18 inst/examples/sandboxed_server/common/binarystream.cpp File Reference

```
#include "binarystream.h"
#include <sstream>
#include <cstring>
#include <memory>
#include <stdexcept>
#include <unistd.h>
#include <sys/types.h>
```

```
#include <sys/socket.h>
#include <sys/un.h>
```

Include dependency graph for binarystream.cpp:



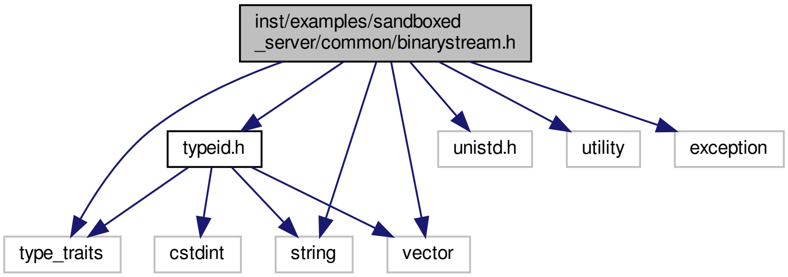
Namespaces

- serialization

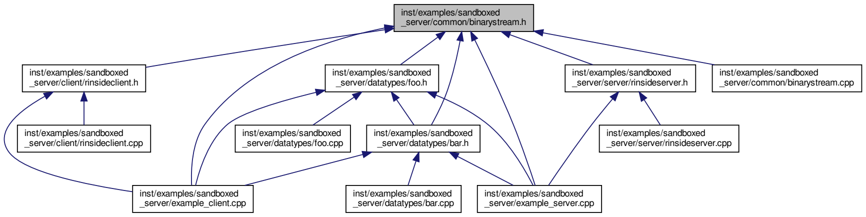
7.19 inst/examples/sandboxed_server/common/binarystream.h File Reference

```
#include "typeid.h"
#include <unistd.h>
#include <string>
#include <vector>
#include <type_traits>
#include <utility>
#include <exception>
```

Include dependency graph for binarystream.h:



This graph shows which files directly or indirectly include this file:



Classes

- class BinaryStream
- class BinaryStream::stream_exception
- struct serialization::serializer< T >
- struct serialization::serializer< std::string >
- struct serialization::serializer< std::vector< T > >
- struct binary_stream_helpers::void_t_struct<... >
- struct binary_stream_helpers::has_serialization_members_cv< T, typename >
- struct binary_stream_helpers::has_serialization_members_cv< T, void_t< serialize_member_t< T >, deserialize_member_t< T > > >
- struct binary_stream_helpers::has_serialization_members< T >

Namespaces

- serialization
- binary_stream_helpers

Typedefs

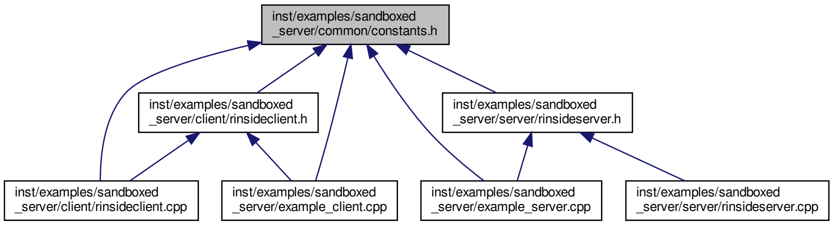
- template<typename... C>
  using binary_stream_helpers::void_t = typename void_t_struct< C... >::type
- template<typename T >
  using binary_stream_helpers::serialize_member_t = decltype(std::declval< T & >().serialize(std::declval< BinaryStream & >()))
- template<typename T >
  using binary_stream_helpers::deserialize_member_t = decltype(T::deserialize(std::declval< BinaryStream & >()))

Functions

- template< typename T >
  std::enable_if< std::is_arithmetic< T >::value >::type binary_stream_helpers::stream_write (BinaryStream &stream, T &t)
- template< typename T >
  std::enable_if< has_typeid< T >::value &&std::is_class< T >::value &&has_serialization_members< T >::value >::type binary_stream_helpers::stream_write (BinaryStream &stream, T &t)
- template< typename T >
  std::enable_if< has_typeid< T >::value &&std::is_class< T >::value &&!has_serialization_members< T >↩ ::value >::type binary_stream_helpers::stream_write (BinaryStream &stream, T &t)
- template< typename T >
  std::enable_if< std::is_arithmetic< T >::value, T >::type binary_stream_helpers::stream_read (BinaryStream &stream)
- template< typename T >
  std::enable_if< has_typeid< T >::value &&std::is_class< T >::value &&has_serialization_members< T >↩ ::value, T >::type binary_stream_helpers::stream_read (BinaryStream &stream)
- template< typename T >
  std::enable_if< has_typeid< T >::value &&std::is_class< T >::value &&!has_serialization_members< T >↩ ::value, T >::type binary_stream_helpers::stream_read (BinaryStream &stream)

7.20 inst/examples/sandboxed_server/common/constants.h File Reference

This graph shows which files directly or indirectly include this file:


inst/examples/sandboxed _server/common/constants.h
inst/examples/sandboxed _server/client/rinsideclient.h
inst/examples/sandboxed _server/server/rinsideserver.h
inst/examples/sandboxed _server/client/rinsideclient.cpp
inst/examples/sandboxed _server/example_client.cpp
inst/examples/sandboxed _server/example_server.cpp
inst/examples/sandboxed _server/server/rinsideserver.cpp

Macros

- #define ris_socket_address "example_server.sock"

Variables

- const uint32_t RIS_MAGIC_NUMBER = 0xF00BA5
- const char RIS_CMD_SETVALUE = 1
- const char RIS_CMD_GETVALUE = 2
- const char RIS_CMD_SETCALLBACK = 3
- const char RIS_CMD_RUN = 4

- const char RIS_CMD_GETCONSOLE = 5
- const char RIS_CMD_INITPLOT = 6
- const char RIS_CMD_GETPLOT = 7
- const char RIS_CMD_EXIT = 8
- const char RIS_REPLY_OK = 101
- const char RIS_REPLY_CALLBACK = 102
- const char RIS_REPLY_VALUE = 103
- const char RIS_REPLY_ERROR = 104

7.20.1 Macro Definition Documentation

7.20.1.1 ris_socket_address

```
#define ris_socket_address "example_server.sock"
```

Definition at line 8 of file constants.h.

Referenced by main(), test_callbacks(), test_console_output(), test_multiple(), test_plot(), and test_setting_getting().

7.20.2 Variable Documentation

7.20.2.1 RIS_CMD_EXIT

```
const char RIS_CMD_EXIT = 8
```

Definition at line 18 of file constants.h.

Referenced by RInsideServer::run(), and RInsideClient::∼RInsideClient().

7.20.2.2 RIS_CMD_GETCONSOLE

```
const char RIS_CMD_GETCONSOLE = 5
```

Definition at line 15 of file constants.h.

Referenced by RInsideClient::getConsoleOutput(), and RInsideServer::run().

7.20.2.3 RIS_CMD_GETPLOT

```
const char RIS_CMD_GETPLOT = 7
```

Definition at line 17 of file constants.h.

Referenced by RInsideClient::getPlot(), and RInsideServer::run().

7.20.2.4 RIS_CMD_GETVALUE

```
const char RIS_CMD_GETVALUE = 2
```

Definition at line 12 of file constants.h.

Referenced by RInsideClient::getValue(), and RInsideServer::run().

7.20.2.5 RIS_CMD_INITPLOT

```
const char RIS_CMD_INITPLOT = 6
```

Definition at line 16 of file constants.h.

Referenced by RInsideClient::initPlot(), and RInsideServer::run().

7.20.2.6 RIS_CMD_RUN

```
const char RIS_CMD_RUN = 4
```

Definition at line 14 of file constants.h.

Referenced by RInsideServer::run(), and RInsideClient::runScript().

7.20.2.7 RIS_CMD_SETCALLBACK

```
const char RIS_CMD_SETCALLBACK = 3
```

Definition at line 13 of file constants.h.

Referenced by RInsideServer::run(), and RInsideClient::setCallback().

7.20.2.8 RIS_CMD_SETVALUE

```
const char RIS_CMD_SETVALUE = 1
```

Definition at line 11 of file constants.h.

Referenced by RInsideServer::run(), and RInsideClient::setValue().

7.20.2.9 RIS_MAGIC_NUMBER

```
const uint32_t RIS_MAGIC_NUMBER = 0xF00BA5
```

Definition at line 10 of file constants.h.

Referenced by RInsideClient::RInsideClient(), and RInsideServer::run().

7.20.2.10 RIS_REPLY_CALLBACK

```
const char RIS_REPLY_CALLBACK = 102
```

Definition at line 22 of file constants.h.

Referenced by CppFunctionForRInsideServer::operator()(), and RInsideClient::runScript().

7.20.2.11 RIS_REPLY_ERROR

```
const char RIS_REPLY_ERROR = 104
```

Definition at line 24 of file constants.h.

Referenced by callback_helper::read_from_stream_with_typeid(), and RInsideClient::readReply().

7.20.2.12 RIS_REPLY_OK

```
const char RIS_REPLY_OK = 101
```

Definition at line 21 of file constants.h.

Referenced by RInsideClient::readReply(), RInsideServer::run(), and RInsideClient::runScript().

7.20.2.13 RIS_REPLY_VALUE

```
const char RIS_REPLY_VALUE = 103
```

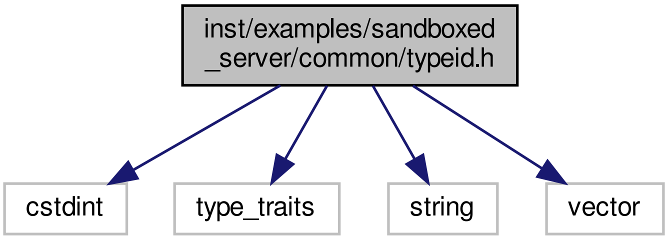
Definition at line 23 of file constants.h.

Referenced by callback_helper::read_from_stream_with_typeid(), RInsideClient::readReply(), RInsideServer::register↩ Type(), RInsideServer::run(), and RInsideClient::runScript().

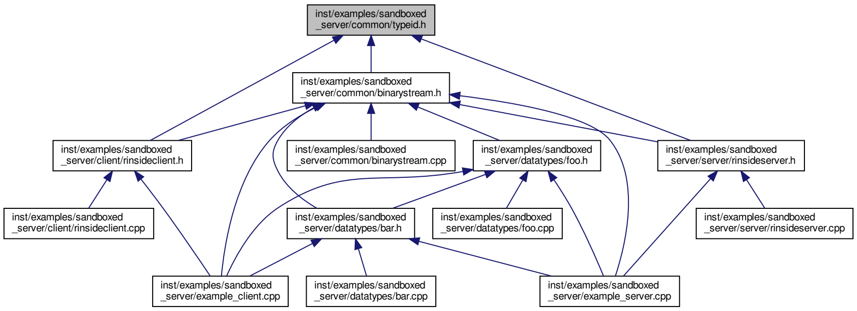
7.21 inst/examples/sandboxed_server/common/typeid.h File Reference

```
#include <cstdint>
#include <type_traits>
#include <string>
#include <vector>
```

Include dependency graph for typeid.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct typeid_helpers::void_t_struct< ... >
- struct typeid_helpers::has_typeid_member< T, typename >
- struct typeid_helpers::has_typeid_member< T, void_t< decltype(T::TYPEID)> >
- struct typeid_helpers::id< T, V >
- struct typeid_helpers::id< T, typename std::enable_if< has_typeid_member< T >::value >::type >
- struct typeid_helpers::id< void, void >
- struct typeid_helpers::id< int8_t, void >
- struct typeid_helpers::id< uint8_t, void >
- struct typeid_helpers::id< int16_t, void >
- struct typeid_helpers::id< uint16_t, void >
- struct typeid_helpers::id< int32_t, void >
- struct typeid_helpers::id< uint32_t, void >
- struct typeid_helpers::id< int64_t, void >
- struct typeid_helpers::id< uint64_t, void >
- struct typeid_helpers::id< float, void >
- struct typeid_helpers::id< double, void >
- struct typeid_helpers::id< std::string, void >
- struct typeid_helpers::id< std::vector< int8_t >, void >
- struct typeid_helpers::id< std::vector< uint8_t >, void >
- struct typeid_helpers::id< std::vector< int16_t >, void >
- struct typeid_helpers::id< std::vector< uint16_t >, void >
- struct typeid_helpers::id< std::vector< int32_t >, void >
- struct typeid_helpers::id< std::vector< uint32_t >, void >
- struct typeid_helpers::id< std::vector< int64_t >, void >
- struct typeid_helpers::id< std::vector< uint64_t >, void >
- struct typeid_helpers::id< std::vector< float >, void >
- struct typeid_helpers::id< std::vector< double >, void >
- struct typeid_helpers::id< std::vector< std::string >, void >
- struct has_typeid< T, typename >
- struct has_typeid< T, typeid_helpers::void_t< decltype(typeid_helpers::id< typename std::decay< T >::type, void >::value)> >

Namespaces

- typeid_helpers

Typedefs

- template<typename... C>
  using typeid_helpers::void_t = typename void_t_struct< C... >::type

Functions

- template<typename T >
  constexpr int32_t TYPEID ()

7.21.1 Function Documentation

7.21.1.1 TYPEID()

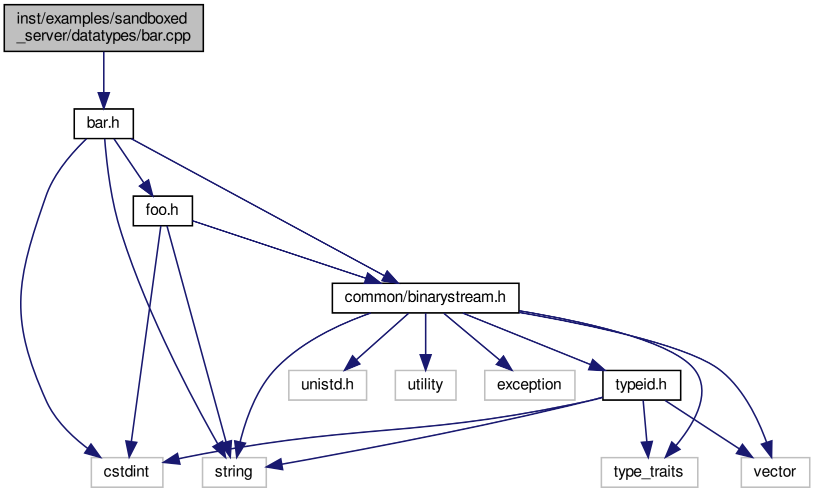
```
template<typename T >
constexpr int32_t TYPEID ( )
```

Definition at line 174 of file typeid.h.

7.22 inst/examples/sandboxed_server/datatypes/bar.cpp File Reference

```
#include "bar.h"
```

Include dependency graph for bar.cpp:

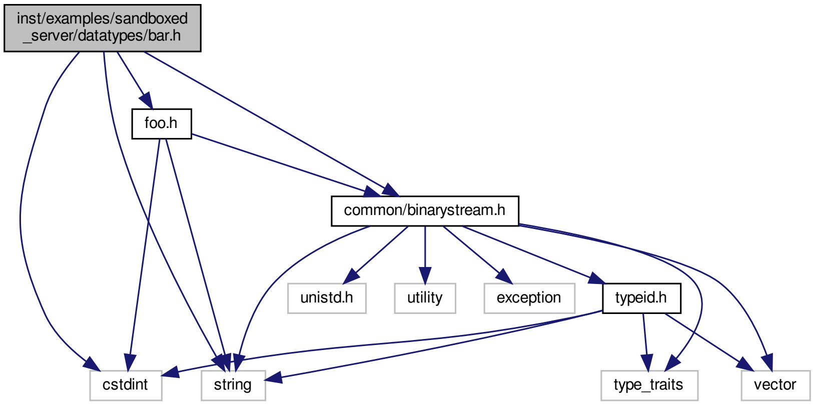


7.23 inst/examples/sandboxed_server/datatypes/bar.h File Reference

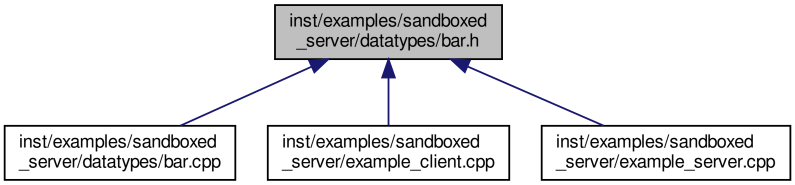
```
#include "common/binarystream.h"
#include <string>
#include <cstdint>
```

```
#include "foo.h"
```

Include dependency graph for bar.h:



This graph shows which files directly or indirectly include this file:

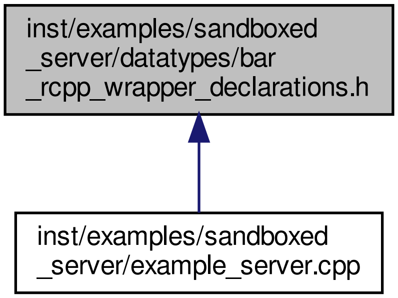


Classes

- class Bar

7.24 inst/examples/sandboxed_server/datatypes/bar_rcpp_wrapper_declarations.h File Reference

This graph shows which files directly or indirectly include this file:



Namespaces

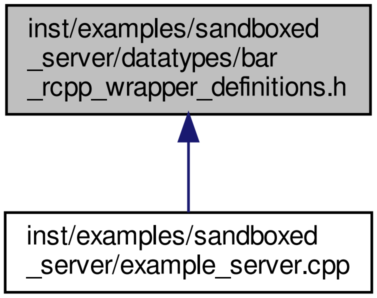
- Rcpp

Functions

- template<>
  SEXP Rcpp::wrap (const Bar &bar)
- template<>
  Bar Rcpp::as (SEXP sexp)

7.25 inst/examples/sandboxed_server/datatypes/bar_rcpp_wrapper_definitions.h File Reference

This graph shows which files directly or indirectly include this file:


inst/examples/sandboxed_server/datatypes/bar_rcpp_wrapper_definitions.h
inst/examples/sandboxed_server/example_server.cpp

Namespaces

- Rcpp

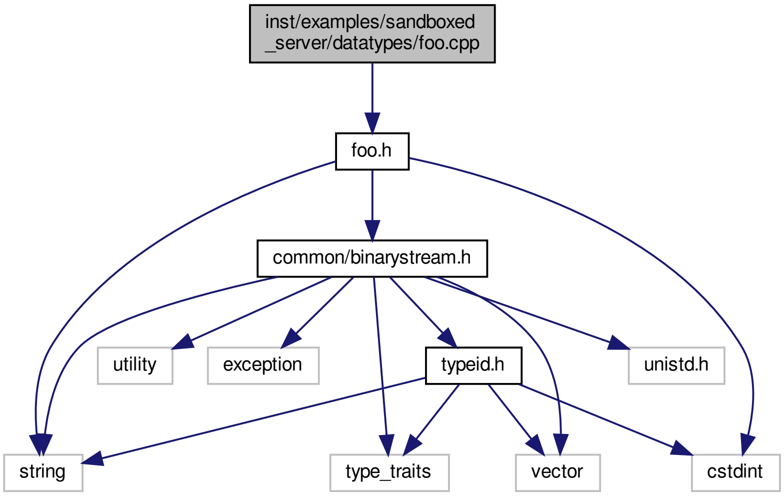
Functions

- template<>
  SEXP Rcpp::wrap (const Bar &bar)
- template<>
  Bar Rcpp::as (SEXP sexp)

7.26 inst/examples/sandboxed_server/datatypes/foo.cpp File Reference

```
#include "foo.h"
```

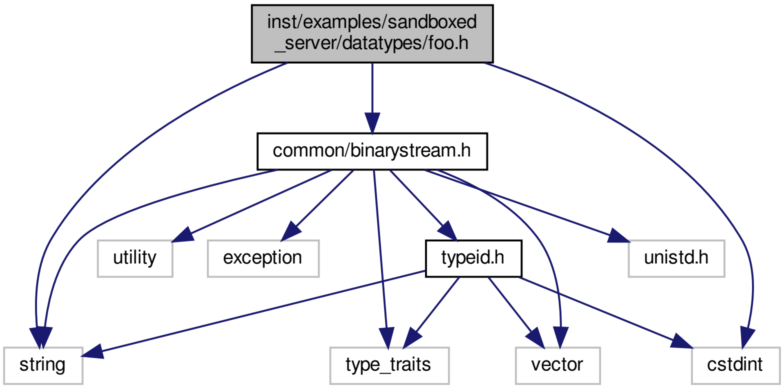
Include dependency graph for foo.cpp:



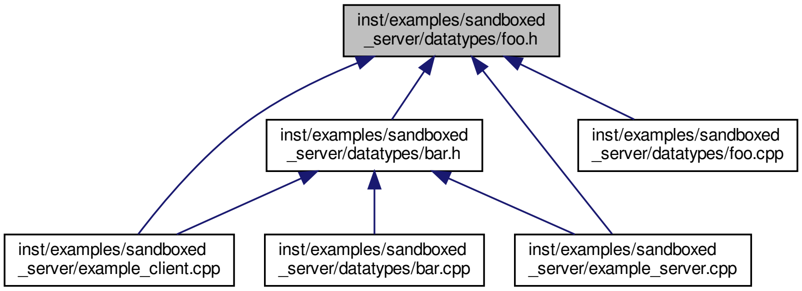
7.27 inst/examples/sandboxed_server/datatypes/foo.h File Reference

```
#include "common/binarystream.h"
#include <string>
#include <cstdint>
```

Include dependency graph for foo.h:



This graph shows which files directly or indirectly include this file:

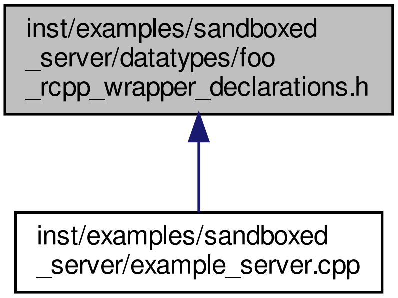


Classes

- class Foo

7.28 inst/examples/sandboxed_server/datatypes/foo_rcpp_wrapper_declarations.h File Reference

This graph shows which files directly or indirectly include this file:



Namespaces

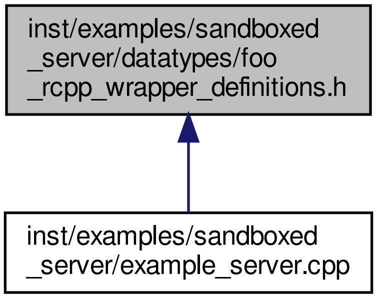
- Rcpp

Functions

- template<>
  SEXP Rcpp::wrap (const Foo &foo)
- template<>
  Foo Rcpp::as (SEXP sexp)

7.29 inst/examples/sandboxed_server/datatypes/foo_rcpp_wrapper_definitions.h File Reference

This graph shows which files directly or indirectly include this file:



Namespaces

- Rcpp

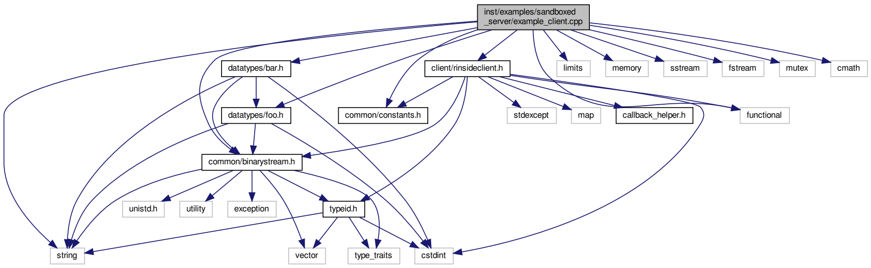
Functions

- template<>
  SEXP Rcpp::wrap (const Foo &foo)
- template<>
  Foo Rcpp::as (SEXP sexp)

7.30 inst/examples/sandboxed_server/example_client.cpp File Reference

```
#include "datatypes/foo.h"
#include "datatypes/bar.h"
#include "common/constants.h"
#include "common/binarystream.h"
#include "client/rinsideclient.h"
#include <limits>
#include <memory>
#include <sstream>
#include <fstream>
#include <string>
#include <mutex>
#include <functional>
#include <cmath>
```

Include dependency graph for example_client.cpp:



Functions

- static void test_setting_getting ()
- static void test_callbacks ()
- static void test_console_output ()
- static void test_plot ()
- static void test_multiple ()
- int main (void)

7.30.1 Function Documentation

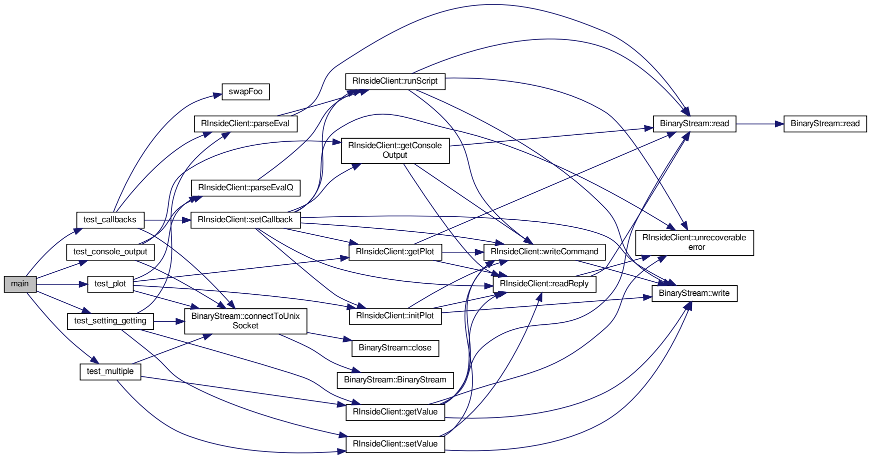
7.30.1.1 main()

```
int main (
            void )
```

Definition at line 200 of file example_client.cpp.

References test_callbacks(), test_console_output(), test_multiple(), test_plot(), and test_setting_getting().

Here is the call graph for this function:



7.30.1.2 test_callbacks()

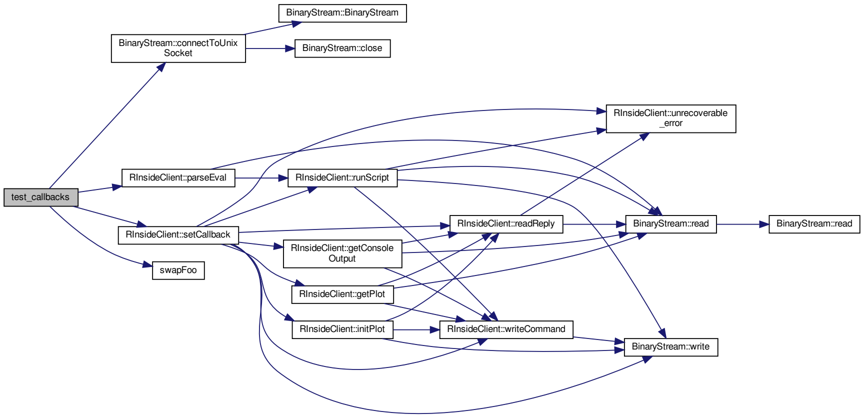
```
static void test_callbacks ( )  [static]
```

Definition at line 64 of file example_client.cpp.

References BinaryStream::connectToUnixSocket(), RInsideClient::parseEval(), ris_socket_address, RInsideClient::setCallback(), and swapFoo().

Referenced by main().

Here is the call graph for this function:



7.30.1.3 test_console_output()

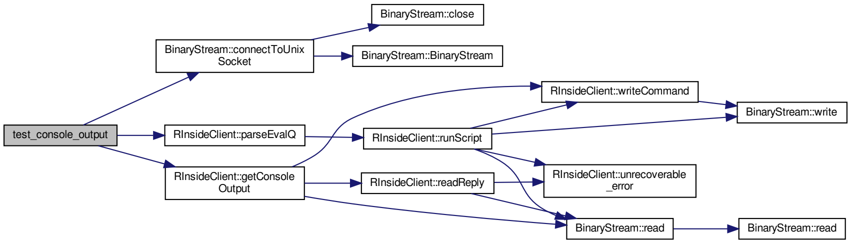
```
static void test_console_output ( )  [static]
```

Definition at line 141 of file example_client.cpp.

References BinaryStream::connectToUnixSocket(), RInsideClient::getConsoleOutput(), RInsideClient::parseEvalQ(), and ris_socket_address.

Referenced by main().

Here is the call graph for this function:


BinaryStream::close
BinaryStream::connectToUnix Socket
BinaryStream::BinaryStream
RInsideClient::writeCommand
BinaryStream::write
test_console_output
RInsideClient::parseEvalQ
RInsideClient::runScript
RInsideClient::unrecoverable _error
RInsideClient::getConsole Output
RInsideClient::readReply
BinaryStream::read
BinaryStream::read

7.30.1.4 test_multiple()

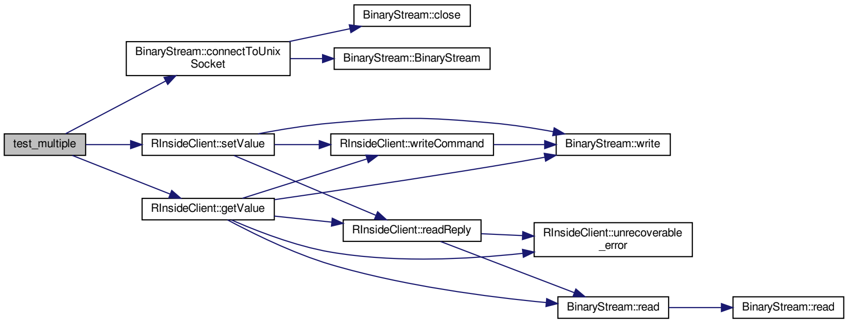
```
static void test_multiple ( )  [static]
```

Definition at line 179 of file example_client.cpp.

References BinaryStream::connectToUnixSocket(), RInsideClient::getValue(), ris_socket_address, and RInsideClient::setValue().

Referenced by main().

Here is the call graph for this function:



7.30.1.5 test_plot()

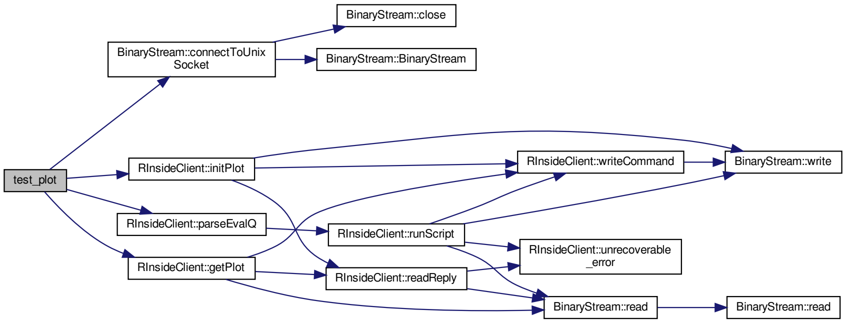
```
static void test_plot ( )  [static]
```

Definition at line 155 of file example_client.cpp.

References BinaryStream::connectToUnixSocket(), RInsideClient::getPlot(), RInsideClient::initPlot(), RInsideClient::parseEvalQ(), and ris_socket_address.

Referenced by main().

Here is the call graph for this function:



7.30.1.6 test_setting_getting()

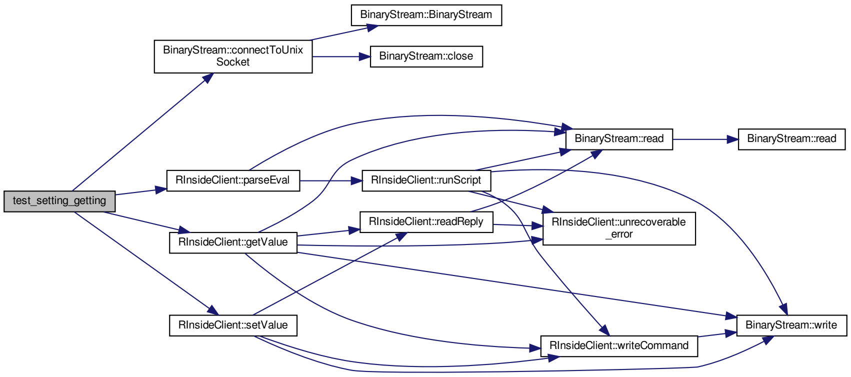
```
static void test_setting_getting ( )  [static]
```

Definition at line 29 of file example_client.cpp.

References Foo::a, Foo::b, BinaryStream::connectToUnixSocket(), RInsideClient::getValue(), Foo::name, RInsideClient::parseEval(), ris_socket_address, and RInsideClient::setValue().

Referenced by main().

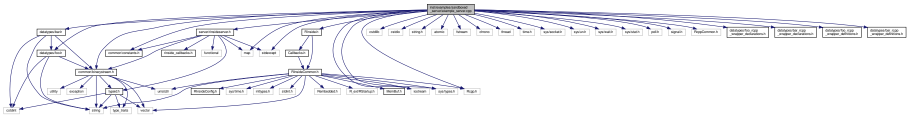
Here is the call graph for this function:



7.31 inst/examples/sandboxed_server/example_server.cpp File Reference

```
#include "common/binarystream.h"
#include "common/constants.h"
#include "datatypes/foo.h"
#include "datatypes/bar.h"
#include <cstdlib>
#include <cstdio>
#include <string.h>
#include <map>
#include <atomic>
#include <iostream>
#include <fstream>
#include <stdexcept>
#include <chrono>
#include <thread>
#include <time.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/un.h>
#include <sys/wait.h>
#include <sys/stat.h>
#include <poll.h>
#include <signal.h>
#include <RcppCommon.h>
#include "datatypes/foo_rcpp_wrapper_declarations.h"
#include "datatypes/bar_rcpp_wrapper_declarations.h"
#include <Rcpp.h>
#include <RInside.h>
#include "datatypes/foo_rcpp_wrapper_definitions.h"
#include "datatypes/bar_rcpp_wrapper_definitions.h"
#include "server/rinsideserver.h"
```

Include dependency graph for example_server.cpp:



Macros

- #define LOG(...) {fprintf(stderr, "%d: ", getpid());fprintf(stderr, __VA_ARGS__);fprintf(stderr, "\n");}

Functions

- int cmpTimespec (const struct timespec &t1, const struct timespec &t2)
- void signal_handler (int signum)
- int main ()

Variables

- const int TIMEOUT_SECONDS = 600

7.31.1 Macro Definition Documentation

7.31.1.1 LOG

```
#define LOG(
              ... ) {fprintf(stderr, "%d:  ", getpid());fprintf(stderr, __VA_ARGS__);fprintf(stderr,
"\n");}
```

Definition at line 42 of file example_server.cpp.

Referenced by main(), CppFunctionForRInsideServer::operator()(), signal_handler(), and RInsideCallbacks::Suicide().

7.31.2 Function Documentation

7.31.2.1 cmpTimespec()

```
int cmpTimespec (
             const struct timespec & t1,
             const struct timespec & t2 )
```

Definition at line 78 of file example_server.cpp.

Referenced by main().

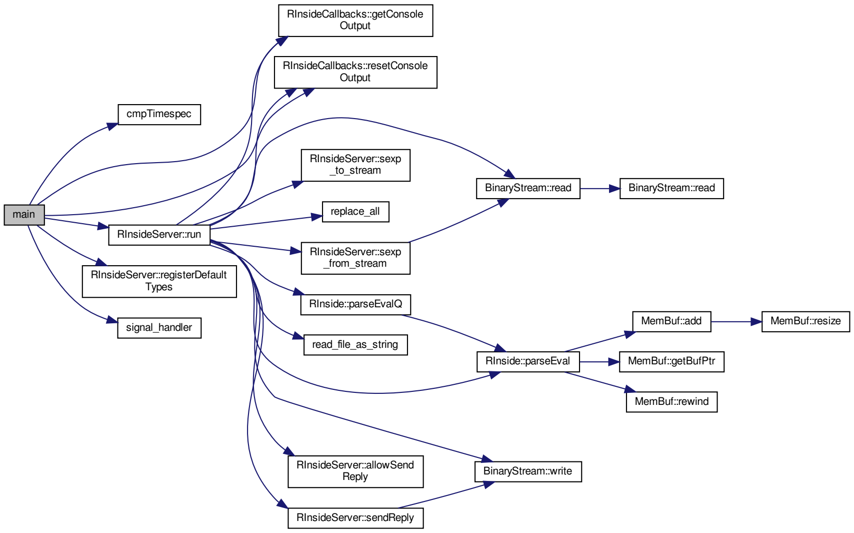
7.31.2.2 main()

```
int main (
            void  )
```

Definition at line 97 of file example_server.cpp.

References cmpTimespec(), RInsideCallbacks::getConsoleOutput(), LOG, RInsideServer::registerDefaultTypes(), R↩ InsideCallbacks::resetConsoleOutput(), ris_socket_address, RInsideServer::run(), signal_handler(), and TIMEOUT_↩ SECONDS.

Here is the call graph for this function:



7.31.2.3 signal_handler()

```
void signal_handler (
            int signum )
```

Definition at line 91 of file example_server.cpp.

References LOG.

Referenced by main().

7.31.3 Variable Documentation

7.31.3.1 TIMEOUT_SECONDS

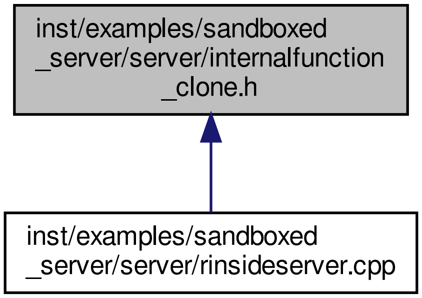
```
const int TIMEOUT_SECONDS = 600
```

Definition at line 52 of file example_server.cpp.

Referenced by main().

7.32 inst/examples/sandboxed_server/server/internalfunction_clone.h File Reference

This graph shows which files directly or indirectly include this file:


inst/examples/sandboxed
_server/server/internalfunction
_clone.h
inst/examples/sandboxed
_server/server/rinsideserver.cpp

Classes

- class CppFunctionForRInsideServer

Namespaces

- Rcpp

Typedefs

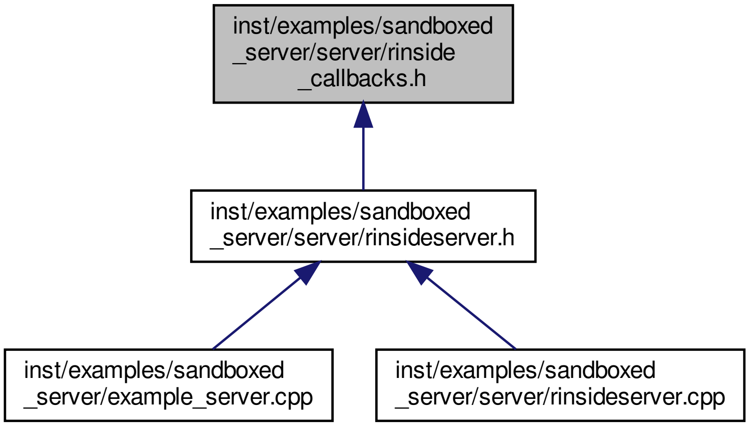
- typedef InternalFunctionForRInsideServer_Impl< PreserveStorage > Rcpp::InternalFunctionForRInsideServer

Functions

- Rcpp::RCPP_API_CLASS (InternalFunctionForRInsideServer_Impl)

7.33 inst/examples/sandboxed_server/server/rinside_callbacks.h File Reference

This graph shows which files directly or indirectly include this file:



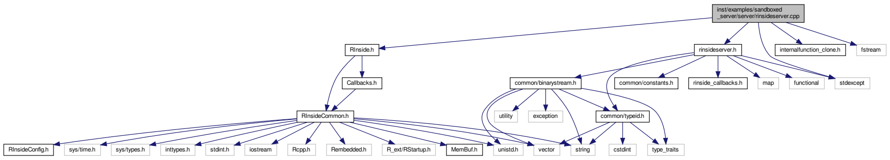
Classes

- class RInsideCallbacks

7.34 inst/examples/sandboxed_server/server/rinsideserver.cpp File Reference

```
#include <RInside.h>
#include "rinsideserver.h"
#include "internalfunction_clone.h"
#include <stdexcept>
#include <fstream>
```

Include dependency graph for rinsideserver.cpp:



Macros

- #define LOG(...) {fprintf(stderr, "%d: ", getpid());fprintf(stderr, __VA_ARGS__);fprintf(stderr, "\n");}
- #define CMD_TRY try {
- #define CMD_CATCH } catch (const BinaryStream::stream_exception) { throw; } catch (const std::exception &e) { std::string s = e.what(); LOG("Command failed: %s", s.c_str()); sendReply(RIS_REPLY_ERROR); stream.↩ write(s); }

Functions

- static void replace_all (std::string &str, const std::string &search, const std::string &replace)
- static std::string read_file_as_string (const std::string &filename)

7.34.1 Macro Definition Documentation

7.34.1.1 CMD_CATCH

```
#define CMD_CATCH } catch (const BinaryStream::stream_exception) { throw; } catch (const std↩
::exception &e) { std::string s = e.what(); LOG("Command failed:  %s", s.c_str()); sendReply(R↩
IS_REPLY_ERROR); stream.write(s); }
```

Definition at line 59 of file rinsideserver.cpp.

Referenced by RInsideServer::run().

7.34.1.2 CMD_TRY

```
#define CMD_TRY try {
```

Definition at line 58 of file rinsideserver.cpp.

Referenced by RInsideServer::run().

7.34.1.3 LOG

```
#define LOG(
              ...  ) {fprintf(stderr, "%d:  ", getpid());fprintf(stderr, __VA_ARGS__);fprintf(stderr,
"\n");}
```

Definition at line 5 of file rinsideserver.cpp.

Referenced by RInsideServer::run(), RInsideServer::sexp_from_stream(), and RInsideServer::sexp_to_stream().

7.34.2 Function Documentation

7.34.2.1 read_file_as_string()

```
static std::string read_file_as_string (
             const std::string & filename )  [static]
```

Definition at line 25 of file rinsideserver.cpp.

References RInsideServer::registry_sexp_from_stream, and RInsideServer::registry_sexp_to_stream.

Referenced by RInsideServer::run().

7.34.2.2 replace_all()

```
static void replace_all (
             std::string & str,
             const std::string & search,
             const std::string & replace )  [static]
```

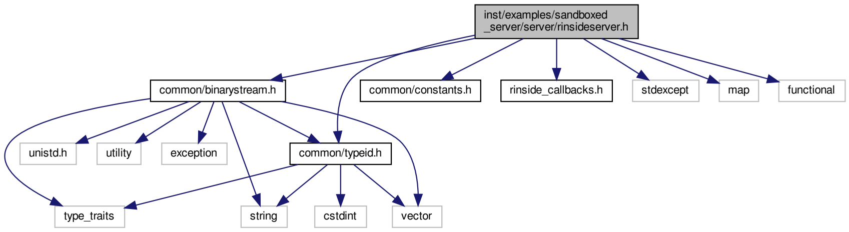
Definition at line 17 of file rinsideserver.cpp.

Referenced by RInsideServer::run().

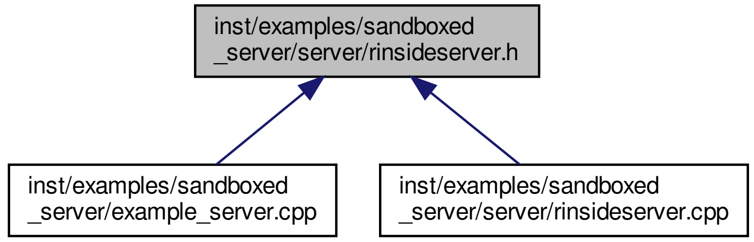
7.35 inst/examples/sandboxed_server/server/rinsideserver.h File Reference

```
#include "common/typeid.h"
#include "common/binarystream.h"
#include "common/constants.h"
#include "rinside_callbacks.h"
#include <stdexcept>
#include <map>
#include <functional>
```

Include dependency graph for rinsideserver.h:



This graph shows which files directly or indirectly include this file:


inst/examples/sandboxed
_server/server/rinsideserver.h
inst/examples/sandboxed
_server/example_server.cpp
inst/examples/sandboxed
_server/server/rinsideserver.cpp

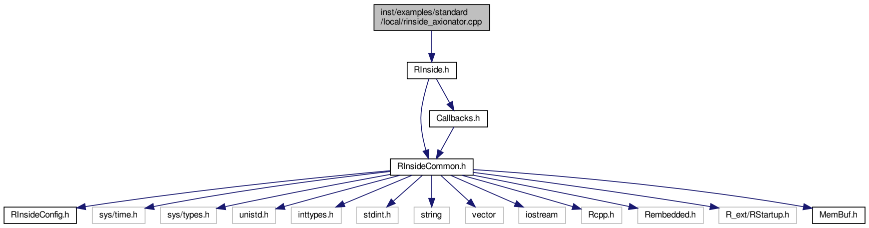
Classes

- class RInsideServer

7.36 inst/examples/standard/local/rinside_axionator.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_axionator.cpp:



Functions

- const char ∗ hello (std::string who)
- int main (int argc, char ∗argv[ ])

7.36.1 Function Documentation

7.36.1.1 hello()

```
const char* hello (
             std::string who )
```

Definition at line 3 of file rinside_axionator.cpp.

Referenced by main().

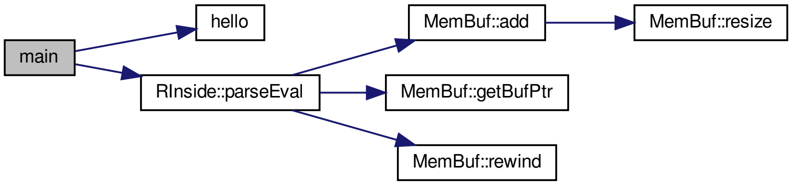
7.36.1.2 main()

```
int main (
             int argc,
             char * argv[] )
```

Definition at line 9 of file rinside_axionator.cpp.

References hello(), and RInside::parseEval().

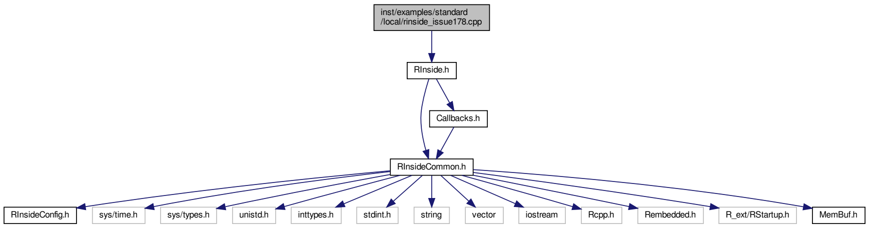
Here is the call graph for this function:



7.37 inst/examples/standard/local/rinside_issue178.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_issue178.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.37.1 Function Documentation

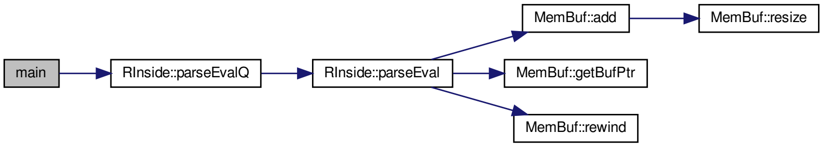
7.37.1.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 7 of file rinside_issue178.cpp.

References RInside::parseEvalQ().

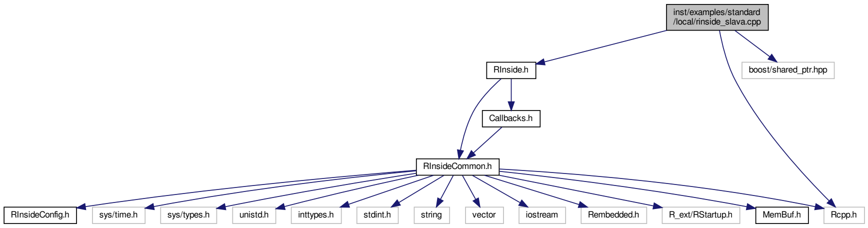
Here is the call graph for this function:



7.38 inst/examples/standard/local/rinside_slava.cpp File Reference

```
#include <RInside.h>
#include <Rcpp.h>
#include <boost/shared_ptr.hpp>
```

Include dependency graph for rinside_slava.cpp:



Functions

- int main (int argc, char ∗argv[ ])

Variables

- static boost::shared_ptr< RInside > Rinst

7.38.1 Function Documentation

7.38.1.1 main()

```
int main (
             int argc,
             char * argv[ ] )
```

Definition at line 9 of file rinside_slava.cpp.

References Rinst.

7.38.2 Variable Documentation

7.38.2.1 Rinst

```
boost::shared_ptr<RInside> Rinst  [static]
```

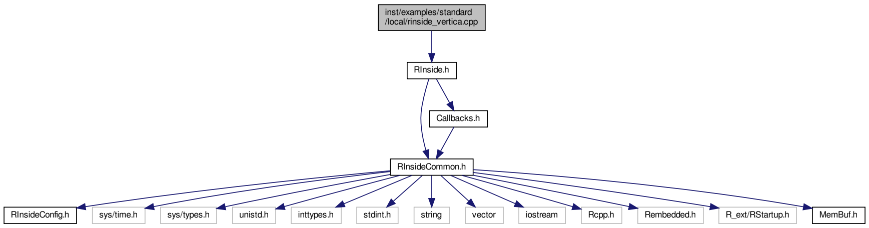
Definition at line 7 of file rinside_slava.cpp.

Referenced by main().

7.39 inst/examples/standard/local/rinside_vertica.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_vertica.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.39.1 Function Documentation

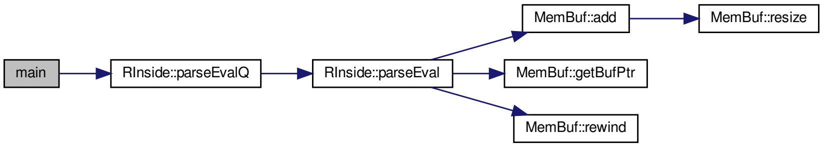
7.39.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 4 of file rinside_vertica.cpp.

References RInside::parseEvalQ().

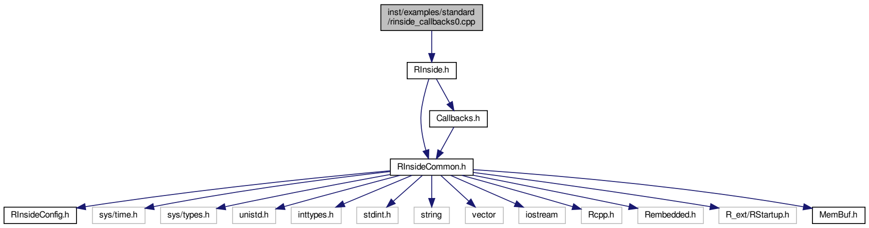
Here is the call graph for this function:



7.40 inst/examples/standard/rinside_callbacks0.cpp File Reference

`#include <RInside.h>`

Include dependency graph for rinside_callbacks0.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.40.1 Function Documentation

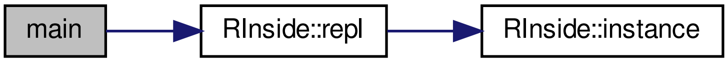
7.40.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 12 of file rinside_callbacks0.cpp.

References RInside::repl().

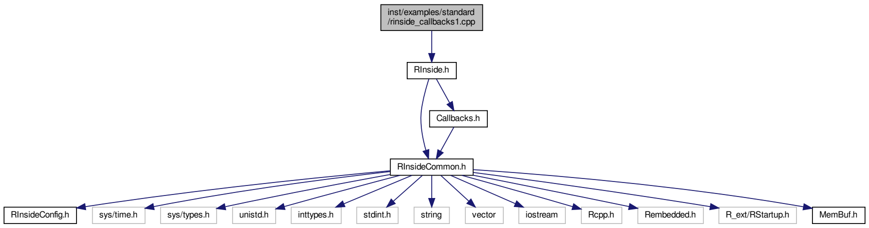
Here is the call graph for this function:



7.41 inst/examples/standard/rinside_callbacks1.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_callbacks1.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.41.1 Function Documentation

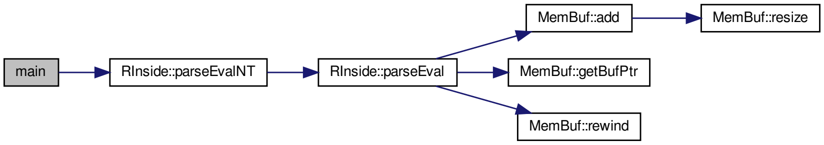
7.41.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 12 of file rinside_callbacks1.cpp.

References RInside::parseEvalNT().

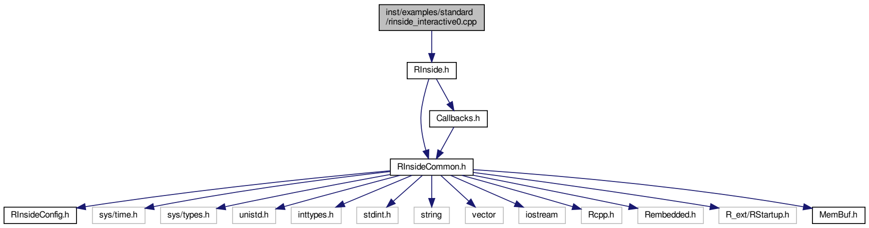
Here is the call graph for this function:



7.42 inst/examples/standard/rinside_interactive0.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_interactive0.cpp:



Classes

- struct Planet
- class Solver
- class Wrapper

Functions

- SEXP Dollar (Rcpp::XPtr< Wrapper > obj, std::string name)
- Rcpp::XPtr< Wrapper > DollarAssign (Rcpp::XPtr< Wrapper > obj, std::string name, SEXP v)
- Rcpp::CharacterVector Names (Rcpp::XPtr< Wrapper > obj)
- int main (int argc, char ∗argv[ ])

7.42.1 Function Documentation

7.42.1.1 Dollar()

```
SEXP Dollar (
             Rcpp::XPtr< Wrapper > obj,
             std::string name )
```

Definition at line 110 of file rinside_interactive0.cpp.

Referenced by main().

7.42.1.2 DollarAssign()

```
Rcpp::XPtr<Wrapper> DollarAssign (
             Rcpp::XPtr< Wrapper > obj,
             std::string name,
             SEXP v )
```

Definition at line 123 of file rinside_interactive0.cpp.

Referenced by main().

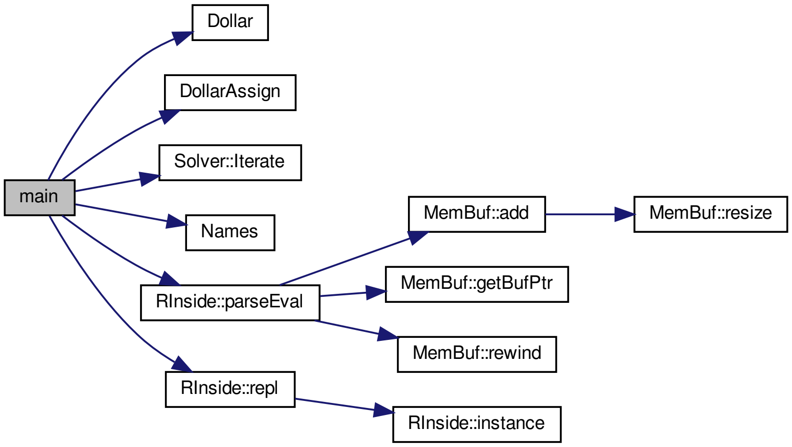
7.42.1.3 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 143 of file rinside_interactive0.cpp.

References Dollar(), DollarAssign(), Solver::Iterate(), Names(), RInside::parseEval(), and RInside::repl().

Here is the call graph for this function:



7.42.1.4 Names()

```
Rcpp::CharacterVector Names (
            Rcpp::XPtr< Wrapper > obj )
```

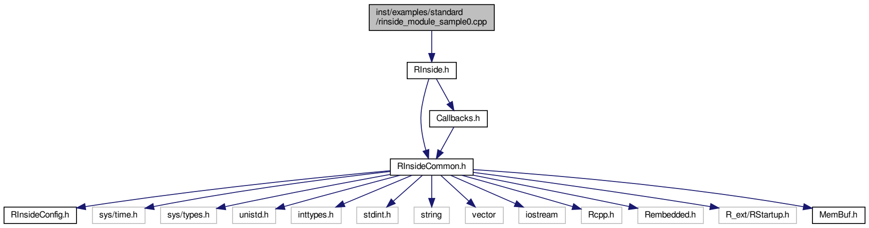
Definition at line 135 of file rinside_interactive0.cpp.

Referenced by main().

7.43 inst/examples/standard/rinside_module_sample0.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_module_sample0.cpp:



Functions

- const char ∗ hello (std::string who)
- int main (int argc, char ∗argv[ ])

7.43.1 Function Documentation

7.43.1.1 hello()

```
const char* hello (
             std::string who )
```

Definition at line 10 of file rinside_module_sample0.cpp.

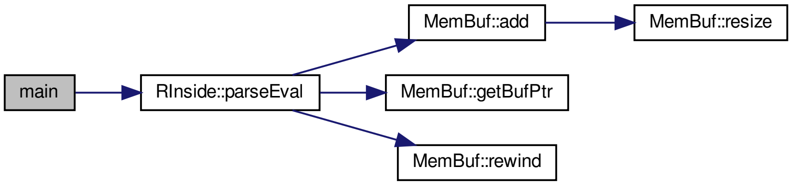
7.43.1.2 main()

```
int main (
             int argc,
             char * argv[] )
```

Definition at line 21 of file rinside_module_sample0.cpp.

References RInside::parseEval().

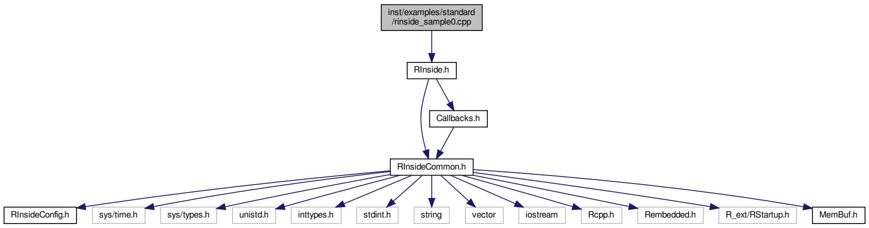
Here is the call graph for this function:



7.44 inst/examples/standard/rinside_sample0.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_sample0.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.44.1 Function Documentation

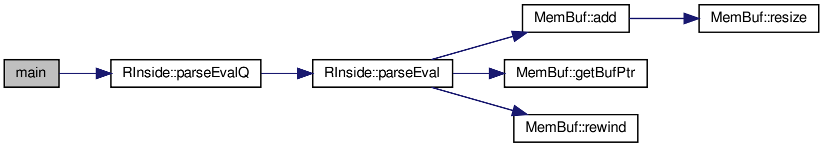
7.44.1.1 main()

```
int main (
              int argc,
              char ∗ argv[ ] )
```

Definition at line 12 of file rinside_sample0.cpp.

References RInside::parseEvalQ().

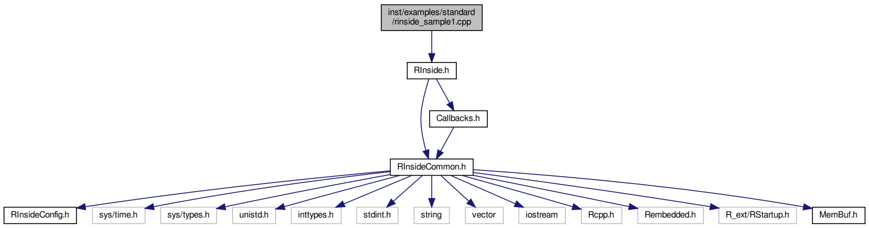
Here is the call graph for this function:



7.45 inst/examples/standard/rinside_sample1.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_sample1.cpp:



Functions

- Rcpp::NumericMatrix createMatrix (const int n)
- int main (int argc, char ∗argv[ ])

7.45.1 Function Documentation

7.45.1.1 createMatrix()

```
Rcpp::NumericMatrix createMatrix (
            const int n )
```

Definition at line 12 of file rinside_sample1.cpp.

Referenced by main().

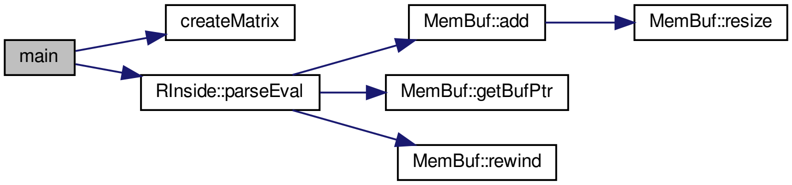
7.45.1.2 main()

```
int main (
            int argc,
            char * argv[] )
```

Definition at line 22 of file rinside_sample1.cpp.

References createMatrix(), and RInside::parseEval().

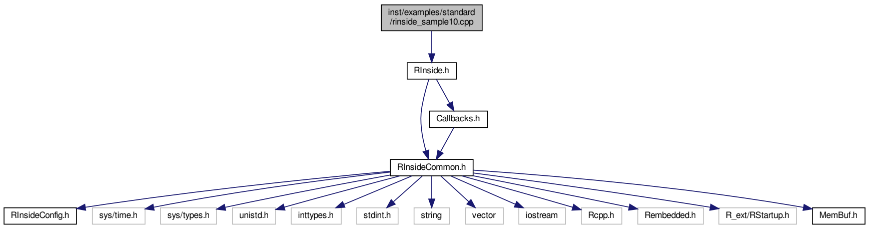
Here is the call graph for this function:



7.46 inst/examples/standard/rinside_sample10.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_sample10.cpp:



Functions

- void show (const Rcpp::List &L)
- int main (int argc, char ∗argv[ ])

7.46.1 Function Documentation

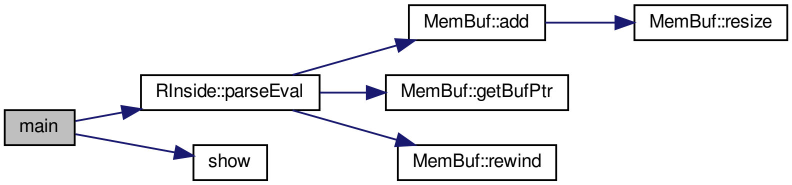
7.46.1.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 18 of file rinside_sample10.cpp.

References RInside::parseEval(), and show().

Here is the call graph for this function:



7.46.1.2 show()

```
void show (
            const Rcpp::List & L )
```

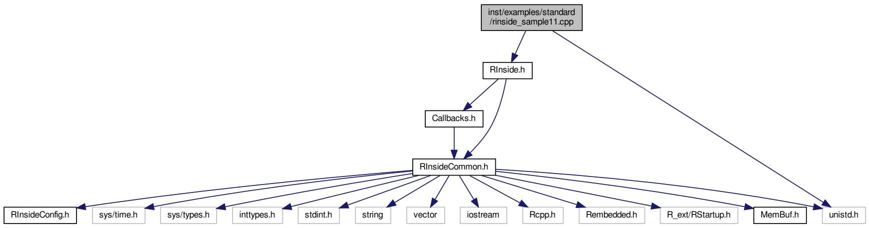
Definition at line 8 of file rinside_sample10.cpp.

Referenced by main().

7.47 inst/examples/standard/rinside_sample11.cpp File Reference

```
#include <RInside.h>
#include <unistd.h>
```

Include dependency graph for rinside_sample11.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.47.1 Function Documentation

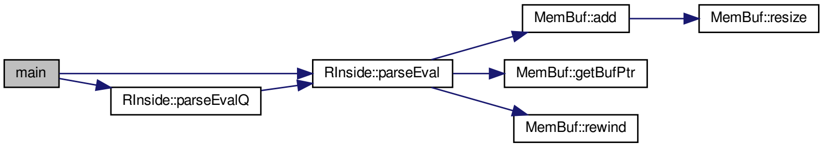
7.47.1.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 9 of file rinside_sample11.cpp.

References RInside::parseEval(), and RInside::parseEvalQ().

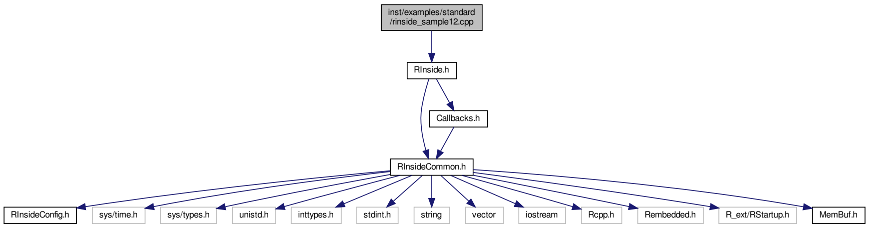
Here is the call graph for this function:



7.48 inst/examples/standard/rinside_sample12.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_sample12.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.48.1 Function Documentation

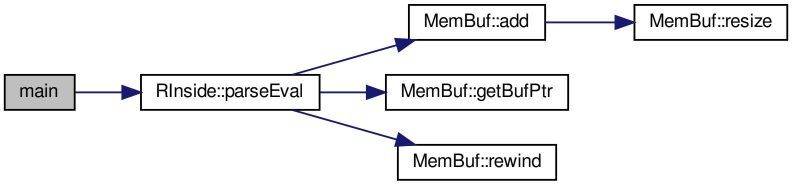
7.48.1.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 9 of file rinside_sample12.cpp.

References RInside::parseEval().

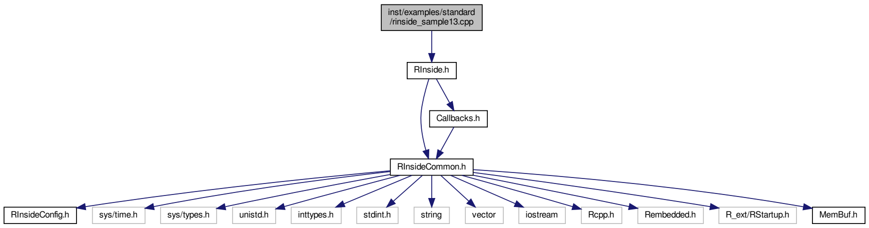
Here is the call graph for this function:



7.49 inst/examples/standard/rinside_sample13.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_sample13.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.49.1 Function Documentation

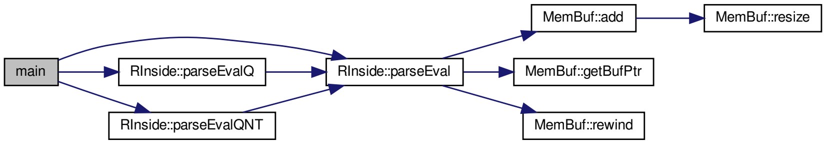
7.49.1.1 main()

```
int main (
             int argc,
             char ∗ argv[ ] )
```

Definition at line 9 of file rinside_sample13.cpp.

References RInside::parseEval(), RInside::parseEvalQ(), and RInside::parseEvalQNT().

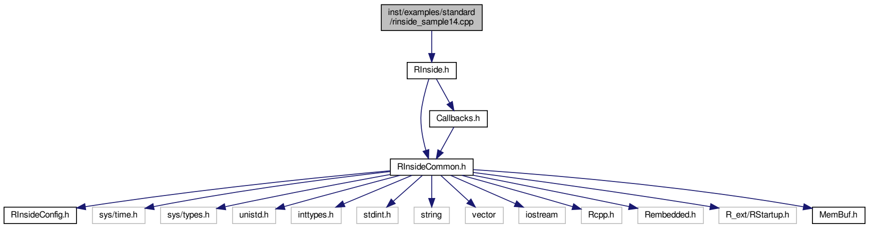
Here is the call graph for this function:



7.50 inst/examples/standard/rinside_sample14.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_sample14.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.50.1 Function Documentation

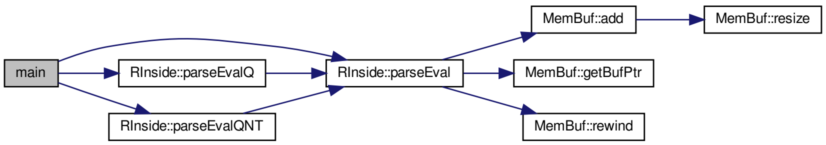
7.50.1.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 9 of file rinside_sample14.cpp.

References RInside::parseEval(), RInside::parseEvalQ(), and RInside::parseEvalQNT().

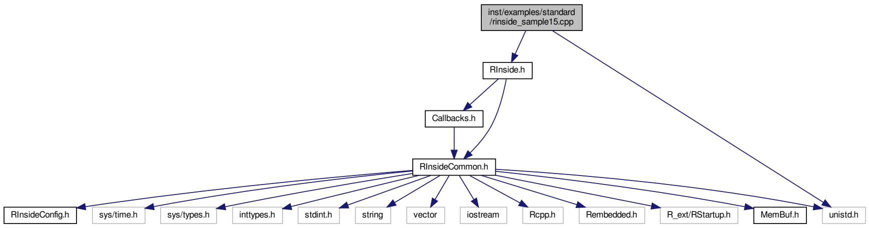
Here is the call graph for this function:



7.51 inst/examples/standard/rinside_sample15.cpp File Reference

```
#include <RInside.h>
#include <unistd.h>
```

Include dependency graph for rinside_sample15.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.51.1 Function Documentation

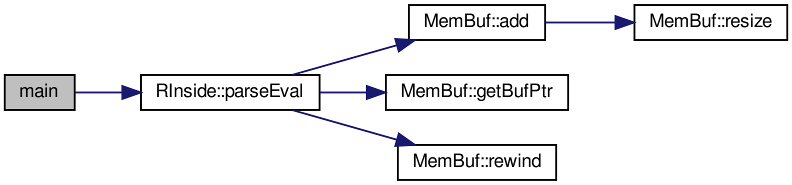
7.51.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 11 of file rinside_sample15.cpp.

References RInside::parseEval().

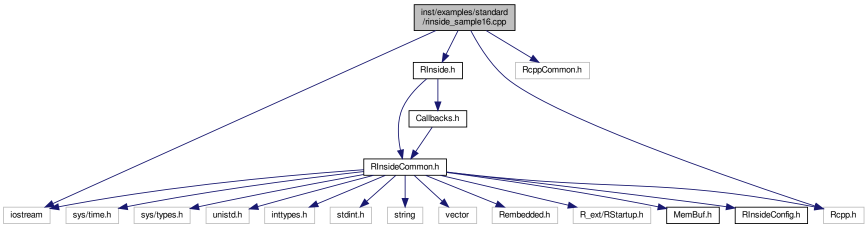
Here is the call graph for this function:



7.52 inst/examples/standard/rinside_sample16.cpp File Reference

```
#include <iostream>
#include <RcppCommon.h>
#include <Rcpp.h>
#include <RInside.h>
```

Include dependency graph for rinside_sample16.cpp:



Classes

- class Foo

Namespaces

- Rcpp

Functions

- template<>
  SEXP Rcpp::wrap (const Foo &foo)
- template<>
  Foo Rcpp::as (SEXP sexp)
- Foo swapFoo (Foo &input)
- int main (int argc, char ∗argv[ ])

7.52.1 Function Documentation

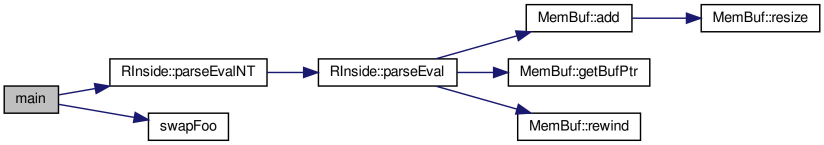
7.52.1.1 main()

```
int main (
              int argc,
              char ∗ argv[ ] )
```

Definition at line 74 of file rinside_sample16.cpp.

References Foo::a, Foo::b, RInside::parseEvalNT(), and swapFoo().

Here is the call graph for this function:



7.52.1.2 swapFoo()

```
Foo swapFoo (
            Foo & input )
```

Definition at line 69 of file rinside_sample16.cpp.

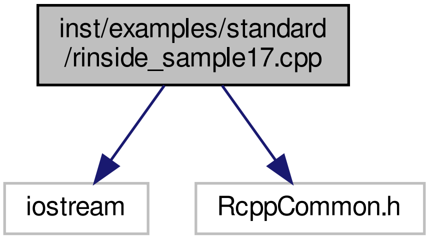
References Foo::a, and Foo::b.

Referenced by main(), and test_callbacks().

7.53 inst/examples/standard/rinside_sample17.cpp File Reference

```
#include <iostream>
#include <RcppCommon.h>
```

Include dependency graph for rinside_sample17.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.53.1 Function Documentation

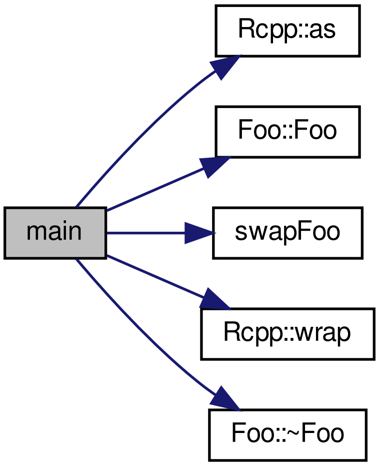
7.53.1.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Definition at line 12 of file rinside_sample17.cpp.

References Foo::a, Rcpp::as(), Foo::b, Foo::Foo(), swapFoo(), Rcpp::wrap(), and Foo::∼Foo().

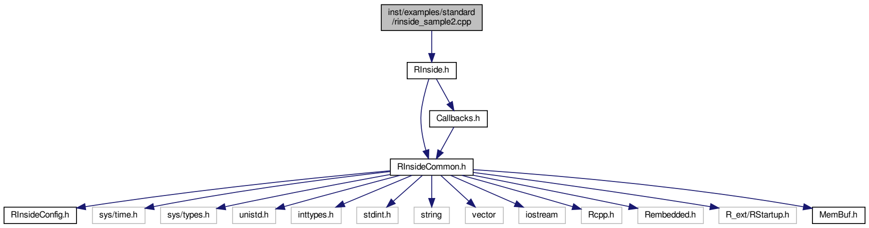
Here is the call graph for this function:



7.54 inst/examples/standard/rinside_sample2.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_sample2.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.54.1 Function Documentation

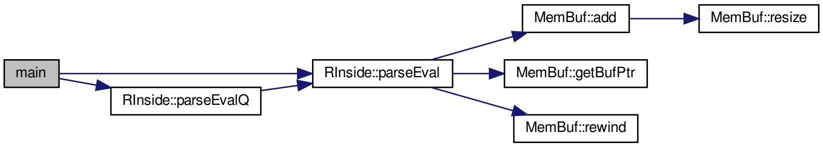
7.54.1.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 10 of file rinside_sample2.cpp.

References RInside::parseEval(), and RInside::parseEvalQ().

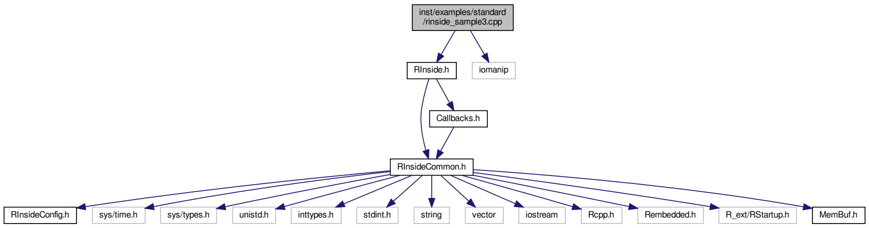
Here is the call graph for this function:



7.55 inst/examples/standard/rinside_sample3.cpp File Reference

```
#include <RInside.h>
#include <iomanip>
```

Include dependency graph for rinside_sample3.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.55.1 Function Documentation

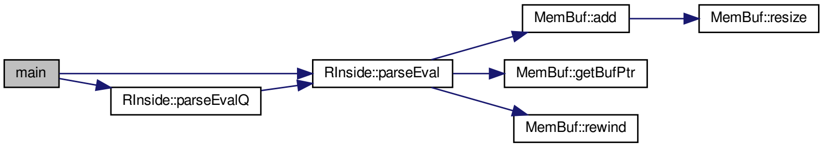
7.55.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 11 of file rinside_sample3.cpp.

References RInside::parseEval(), and RInside::parseEvalQ().

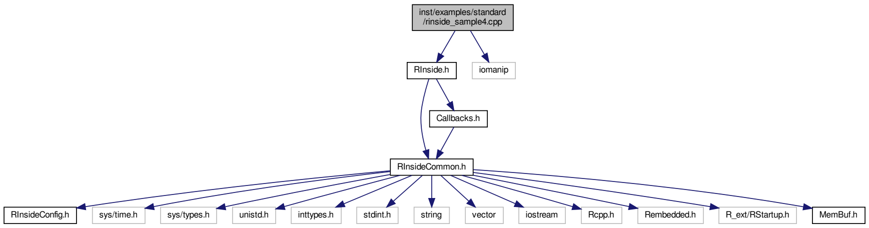
Here is the call graph for this function:



7.56 inst/examples/standard/rinside_sample4.cpp File Reference

```
#include <RInside.h>
#include <iomanip>
```

Include dependency graph for rinside_sample4.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.56.1 Function Documentation

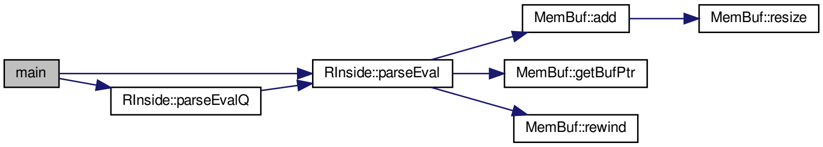
7.56.1.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 11 of file rinside_sample4.cpp.

References RInside::parseEval(), and RInside::parseEvalQ().

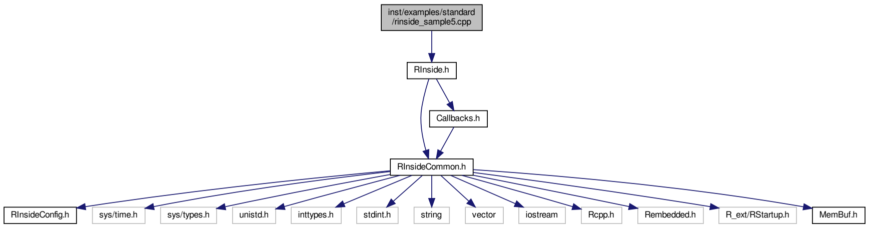
Here is the call graph for this function:



7.57 inst/examples/standard/rinside_sample5.cpp File Reference

`#include <RInside.h>`

Include dependency graph for rinside_sample5.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.57.1 Function Documentation

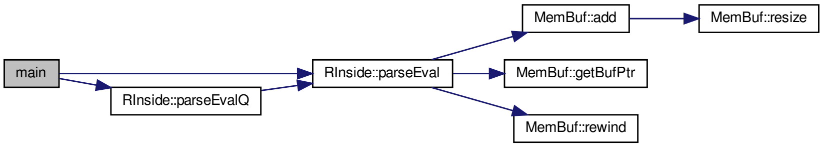
7.57.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 10 of file rinside_sample5.cpp.

References RInside::parseEval(), and RInside::parseEvalQ().

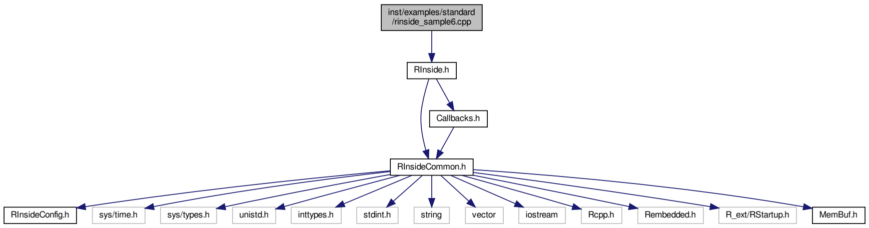
Here is the call graph for this function:



7.58 inst/examples/standard/rinside_sample6.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_sample6.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.58.1 Function Documentation

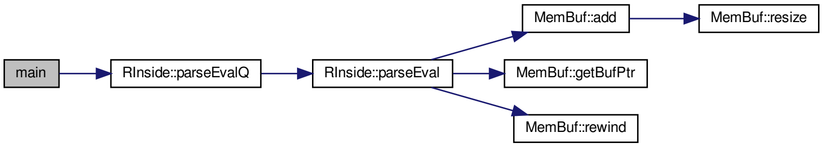
7.58.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 10 of file rinside_sample6.cpp.

References RInside::parseEvalQ().

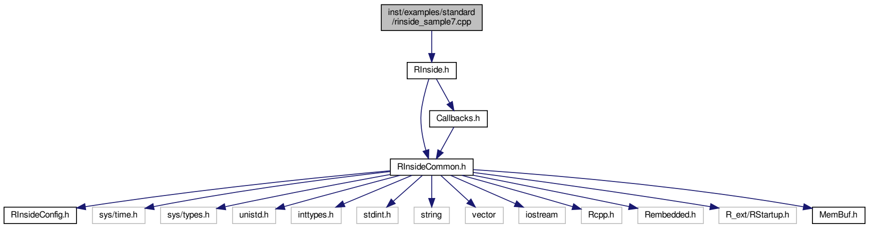
Here is the call graph for this function:



7.59 inst/examples/standard/rinside_sample7.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_sample7.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.59.1 Function Documentation

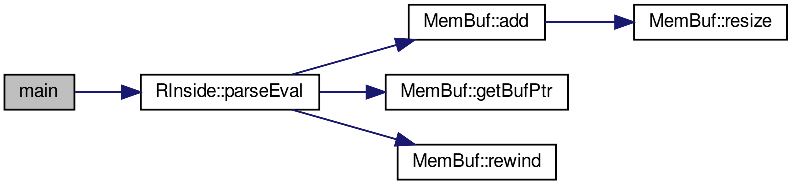
7.59.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 9 of file rinside_sample7.cpp.

References RInside::parseEval().

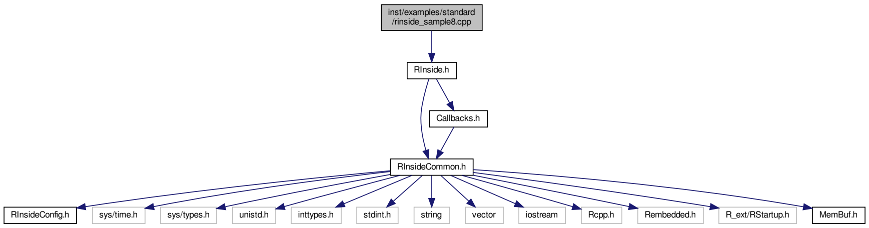
Here is the call graph for this function:



7.60 inst/examples/standard/rinside_sample8.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_sample8.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.60.1 Function Documentation

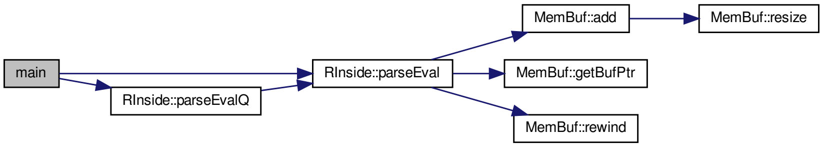
7.60.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 9 of file rinside_sample8.cpp.

References RInside::parseEval(), and RInside::parseEvalQ().

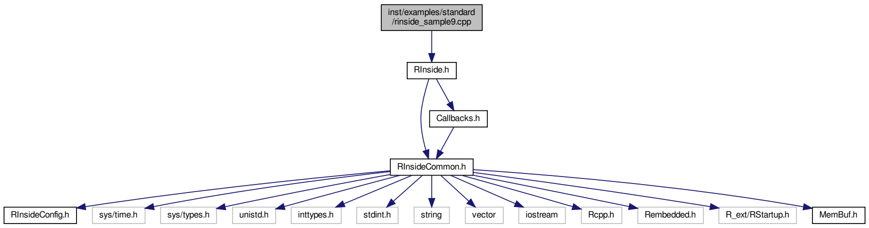
Here is the call graph for this function:



7.61 inst/examples/standard/rinside_sample9.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_sample9.cpp:



Functions

- std::string hello (std::string who)
- int main (int argc, char ∗argv[ ])

7.61.1 Function Documentation

7.61.1.1 hello()

```
std::string hello (
              std::string who )
```

Definition at line 10 of file rinside_sample9.cpp.

Referenced by main().

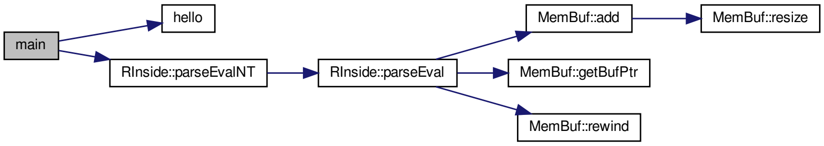
7.61.1.2 main()

```
int main (
              int argc,
              char * argv[ ] )
```

Definition at line 16 of file rinside_sample9.cpp.

References hello(), and RInside::parseEvalNT().

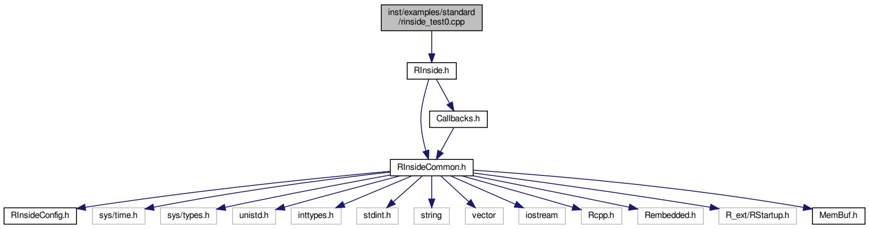
Here is the call graph for this function:



7.62 inst/examples/standard/rinside_test0.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_test0.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.62.1 Function Documentation

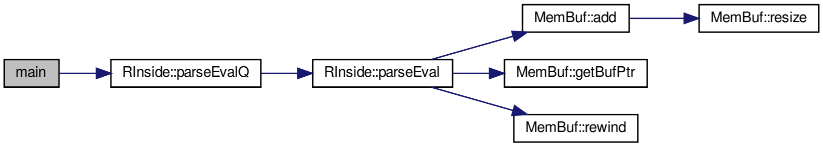
7.62.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 9 of file rinside_test0.cpp.

References RInside::parseEvalQ().

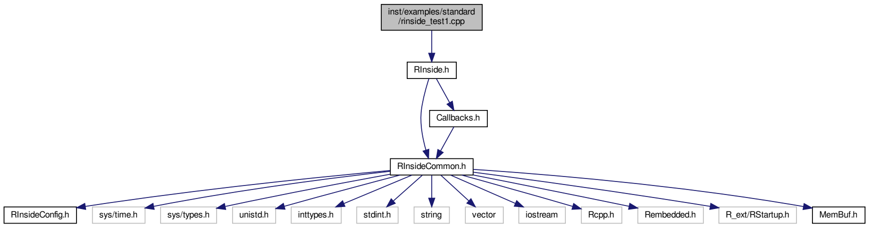
Here is the call graph for this function:



7.63 inst/examples/standard/rinside_test1.cpp File Reference

`#include <RInside.h>`

Include dependency graph for rinside_test1.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.63.1 Function Documentation

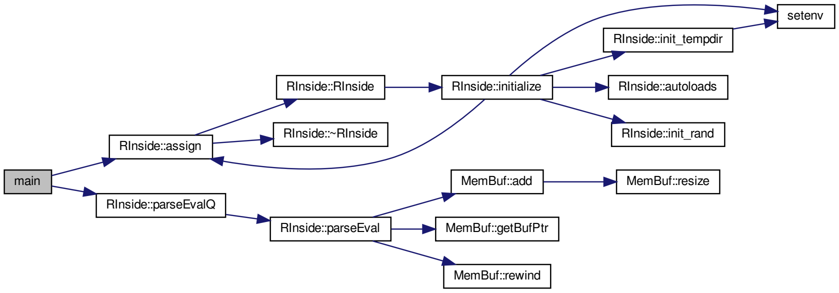
7.63.1.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 9 of file rinside_test1.cpp.

References RInside::assign(), and RInside::parseEvalQ().

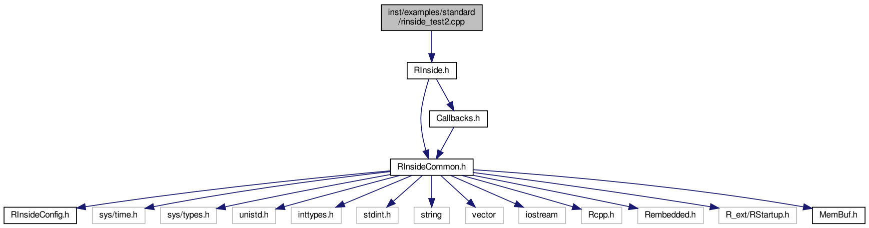
Here is the call graph for this function:



7.64 inst/examples/standard/rinside_test2.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for rinside_test2.cpp:



Functions

- int main (int argc, char ∗argv[ ])

7.64.1 Function Documentation

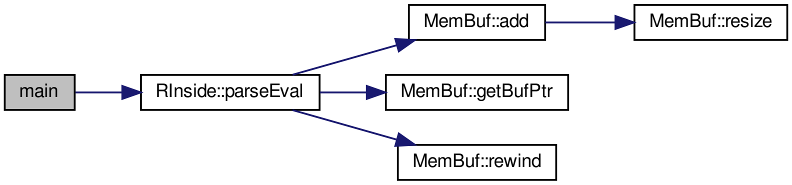
7.64.1.1 main()

```
int main (
            int argc,
            char ∗ argv[ ] )
```

Definition at line 9 of file rinside_test2.cpp.

References RInside::parseEval().

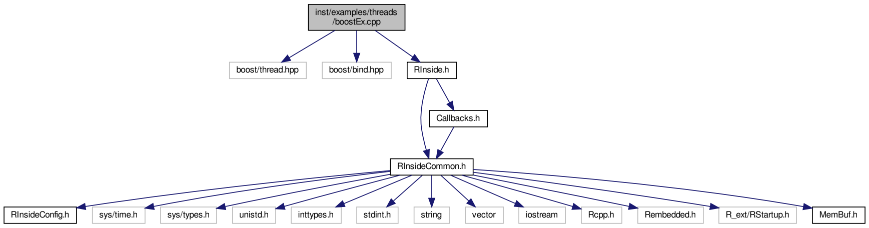
Here is the call graph for this function:



7.65 inst/examples/threads/boostEx.cpp File Reference

```
#include <boost/thread.hpp>
#include <boost/bind.hpp>
#include <RInside.h>
```

Include dependency graph for boostEx.cpp:



Classes

- class Resource

Functions

- void thread_func (Resource &resource)
- int main (int argc, char ∗argv[ ])

Variables

- uintptr_t R_CStackLimit

7.65.1 Function Documentation

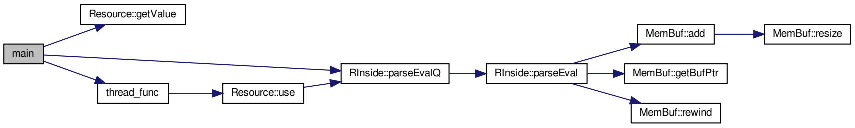
7.65.1.1 main()

```
int main (
             int argc,
             char ∗ argv[ ] )
```

Definition at line 31 of file boostEx.cpp.

References Resource::getValue(), RInside::parseEvalQ(), and thread_func().

Here is the call graph for this function:



7.65.1.2 thread_func()

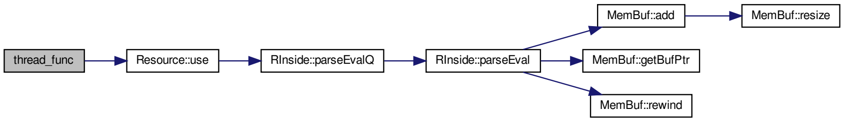
```
void thread_func (
            Resource & resource )
```

Definition at line 25 of file boostEx.cpp.

References R_CStackLimit, and Resource::use().

Referenced by main().

Here is the call graph for this function:



7.65.2 Variable Documentation

7.65.2.1 R_CStackLimit

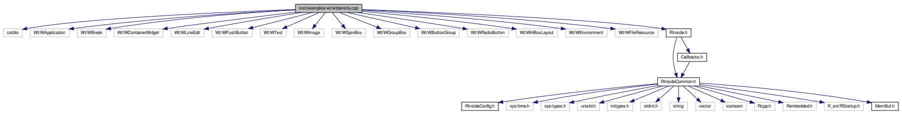
```
uintptr_t R_CStackLimit
```

Referenced by RInside::initialize(), and thread_func().

7.66 inst/examples/wt/wtdensity.cpp File Reference

```
#include <cstdio>
#include <Wt/WApplication>
#include <Wt/WBreak>
#include <Wt/WContainerWidget>
#include <Wt/WLineEdit>
#include <Wt/WPushButton>
#include <Wt/WText>
#include <Wt/WImage>
#include <Wt/WSpinBox>
#include <Wt/WGroupBox>
#include <Wt/WButtonGroup>
#include <Wt/WRadioButton>
#include <Wt/WHBoxLayout>
#include <Wt/WEnvironment>
#include <Wt/WFileResource>
#include <RInside.h>
```

Include dependency graph for wtdensity.cpp:



Classes

- class DensityApp

Functions

- WApplication ∗ createApplication (const WEnvironment &env)
- int main (int argc, char ∗∗argv)

7.66.1 Function Documentation

7.66.1.1 createApplication()

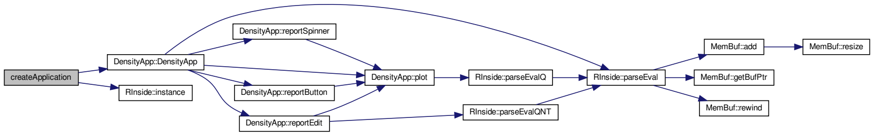
```
WApplication* createApplication (
             const WEnvironment & env )
```

Definition at line 184 of file wtdensity.cpp.

References DensityApp::DensityApp(), and RInside::instance().

Referenced by main().

Here is the call graph for this function:



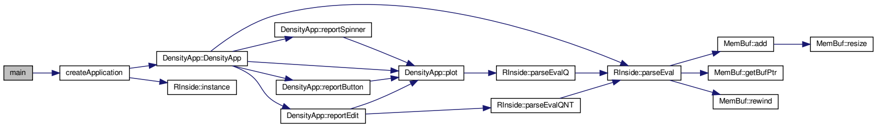
7.66.1.2 main()

```
int main (
             int argc,
             char ** argv )
```

Definition at line 194 of file wtdensity.cpp.

References createApplication().

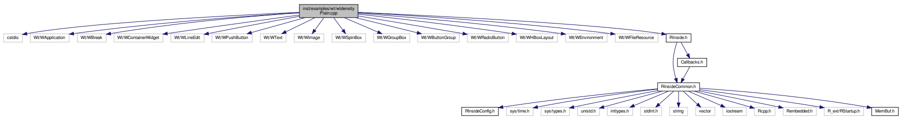
Here is the call graph for this function:



7.67 inst/examples/wt/wtdensityPlain.cpp File Reference

```
#include <cstdio>
#include <Wt/WApplication>
#include <Wt/WBreak>
#include <Wt/WContainerWidget>
#include <Wt/WLineEdit>
#include <Wt/WPushButton>
#include <Wt/WText>
#include <Wt/WImage>
#include <Wt/WSpinBox>
#include <Wt/WGroupBox>
#include <Wt/WButtonGroup>
#include <Wt/WRadioButton>
#include <Wt/WHBoxLayout>
#include <Wt/WEnvironment>
#include <Wt/WFileResource>
#include <RInside.h>
```

Include dependency graph for wtdensityPlain.cpp:



Classes

- class DensityApp

Functions

- WApplication ∗ createApplication (const WEnvironment &env)
- int main (int argc, char ∗∗argv)

7.67.1 Function Documentation

7.67.1.1 createApplication()

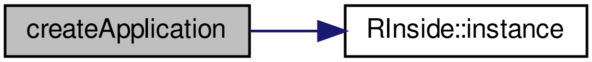
```
WApplication* createApplication (
             const WEnvironment & env )
```

Definition at line 171 of file wtdensityPlain.cpp.

References RInside::instance().

Referenced by main().

Here is the call graph for this function:



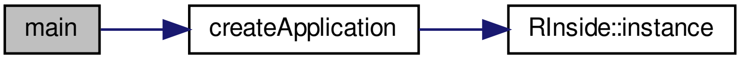
7.67.1.2 main()

```
int main (
             int argc,
             char ** argv )
```

Definition at line 181 of file wtdensityPlain.cpp.

References createApplication().

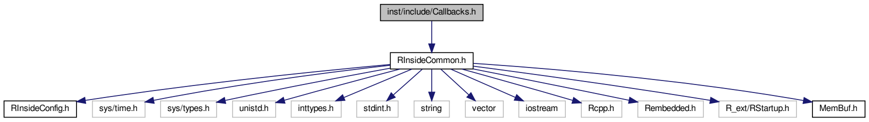
Here is the call graph for this function:



7.68 inst/include/Callbacks.h File Reference

```
#include <RInsideCommon.h>
```

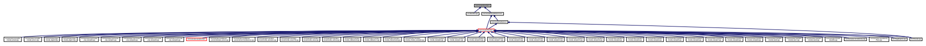
Include dependency graph for Callbacks.h:



This graph shows which files directly or indirectly include this file:

7.69 inst/include/MemBuf.h File Reference

This graph shows which files directly or indirectly include this file:



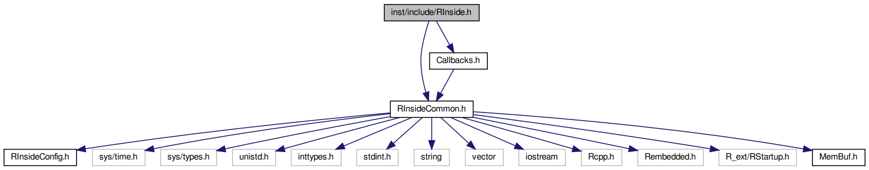
Classes

- class MemBuf

7.70 inst/include/RInside.h File Reference

```
#include <RInsideCommon.h>
#include <Callbacks.h>
```

Include dependency graph for RInside.h:



This graph shows which files directly or indirectly include this file:

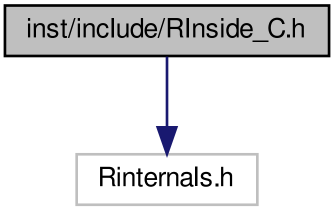
Classes

- class RInside
- class RInside::Proxy

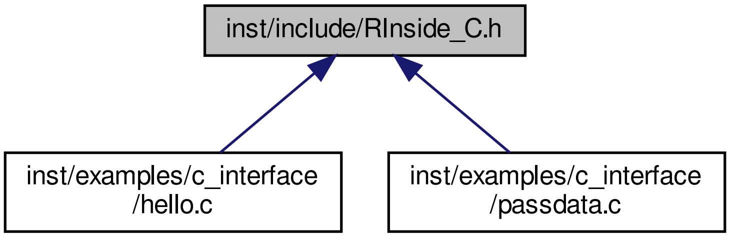
7.71 inst/include/RInside_C.h File Reference

```
#include <Rinternals.h>
```

Include dependency graph for RInside_C.h:



This graph shows which files directly or indirectly include this file:


inst/include/RInside_C.h
inst/examples/c_interface /hello.c
inst/examples/c_interface /passdata.c

Functions

- void setupRinC ()
- void passToR (SEXP x, char ∗name)
- SEXP evalInR (char ∗cmd)
- void evalQuietlyInR (char ∗cmd)
- void teardownRinC ()

7.71.1 Function Documentation

7.71.1.1 evalInR()

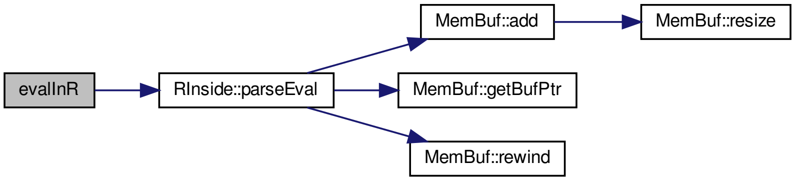
```
SEXP evalInR (
             char * cmd )
```

Definition at line 36 of file RInside_C.cpp.

References RInside::parseEval().

Referenced by main().

Here is the call graph for this function:



7.71.1.2 evalQuietlyInR()

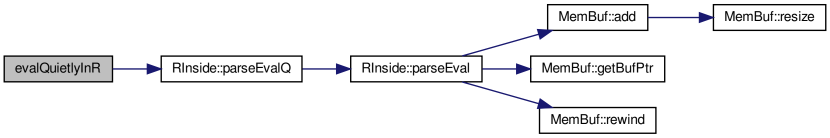
```
void evalQuietlyInR (
             char * cmd )
```

Definition at line 43 of file RInside_C.cpp.

References RInside::parseEvalQ().

Referenced by main().

Here is the call graph for this function:



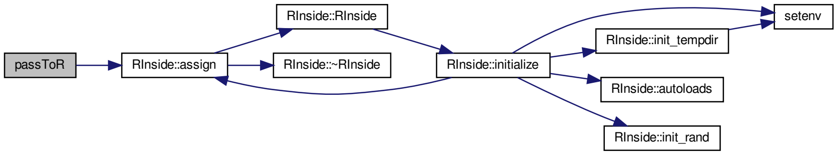
7.71.1.3 passToR()

```
void passToR (
             SEXP x,
             char * name )
```

Definition at line 31 of file RInside_C.cpp.

References RInside::assign().

Here is the call graph for this function:



7.71.1.4 setupRinC()

```
void setupRinC ( )
```

Definition at line 26 of file RInside_C.cpp.

Referenced by main().

7.71.1.5 teardownRinC()

```
void teardownRinC ( )
```

Definition at line 48 of file RInside_C.cpp.

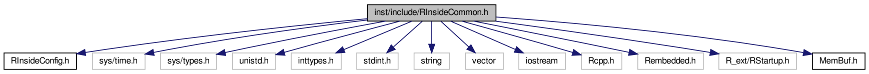
References rr.

Referenced by main().

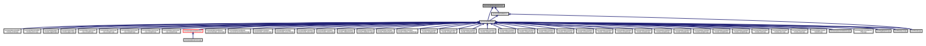
7.72 inst/include/RInsideCommon.h File Reference

```
#include <RInsideConfig.h>
#include <sys/time.h>
#include <sys/types.h>
#include <unistd.h>
#include <inttypes.h>
#include <stdint.h>
#include <string>
#include <vector>
#include <iostream>
#include <Rcpp.h>
#include <Rembedded.h>
#include <R_ext/RStartup.h>
#include <MemBuf.h>
```

Include dependency graph for RInsideCommon.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define CSTACK_DEFNS
- #define HAVE_UINTPTR_T
- #define logTxt(x, b)

Functions

- void logTxtFunction (const char ∗file, const int line, const char ∗expression, const bool verbose)

7.72.1 Macro Definition Documentation

7.72.1.1 CSTACK_DEFNS

```
#define CSTACK_DEFNS
```

Definition at line 49 of file RInsideCommon.h.

7.72.1.2 HAVE_UINTPTR_T

```
#define HAVE_UINTPTR_T
```

Definition at line 51 of file RInsideCommon.h.

7.72.1.3 logTxt

```
#define logTxt(
            x,
            b )
```

Definition at line 71 of file RInsideCommon.h.

7.72.2 Function Documentation

7.72.2.1 logTxtFunction()

```
void logTxtFunction (
            const char * file,
            const int line,
            const char * expression,
            const bool verbose )  [inline]
```

Definition at line 61 of file RInsideCommon.h.

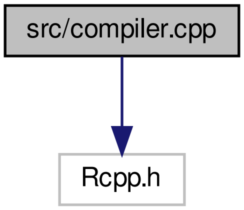
7.73 inst/include/RInsideConfig.h File Reference

This graph shows which files directly or indirectly include this file:

7.74 src/compiler.cpp File Reference

```
#include <Rcpp.h>
```

Include dependency graph for compiler.cpp:



Functions

- void showCompiler ()

7.74.1 Function Documentation

7.74.1.1 showCompiler()

```
void showCompiler ( )
```

Definition at line 5 of file compiler.cpp.

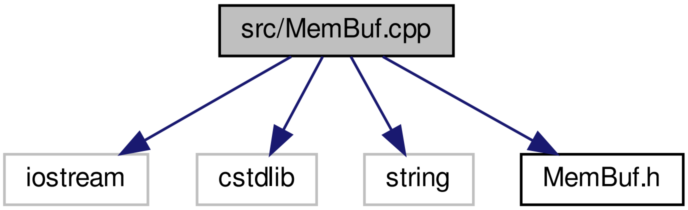
Referenced by _RInside_showCompiler().

7.75 src/MemBuf.cpp File Reference

```
#include <iostream>
#include <cstdlib>
#include <string>
```

```
#include <MemBuf.h>
```

Include dependency graph for MemBuf.cpp:



Variables

- bool verbose
- const char ∗ programName

7.75.1 Variable Documentation

7.75.1.1 programName

```
const char* programName
```

Definition at line 32 of file RInside.cpp.

Referenced by RInside::initialize(), and RInside::parseEval().

7.75.1.2 verbose

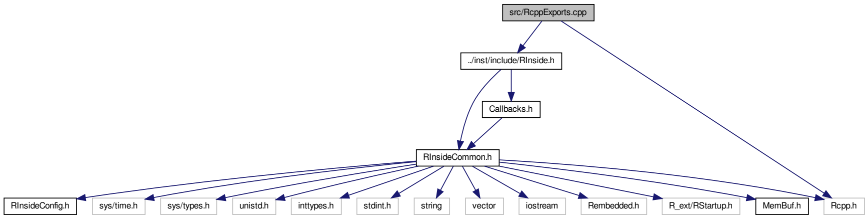
```
bool verbose
```

Referenced by RInside::assign(), RInside::initialize(), and RInside::setVerbose().

7.76 src/RcppExports.cpp File Reference

```
#include "../inst/include/RInside.h"
#include <Rcpp.h>
```

Include dependency graph for RcppExports.cpp:



Functions

- void showCompiler ()
- RcppExport SEXP _RInside_showCompiler ()
- RcppExport void R_init_RInside (DllInfo ∗dll)

Variables

- static const R_CallMethodDef CallEntries [ ]

7.76.1 Function Documentation

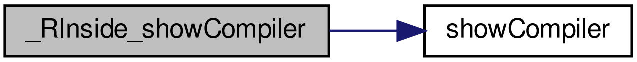
7.76.1.1 _RInside_showCompiler()

```
RcppExport SEXP _RInside_showCompiler ( )
```

Definition at line 11 of file RcppExports.cpp.

References showCompiler().

Here is the call graph for this function:



7.76.1.2 R_init_RInside()

```
RcppExport void R_init_RInside (
             DllInfo * dll )
```

Definition at line 24 of file RcppExports.cpp.

References CallEntries.

7.76.1.3 showCompiler()

```
void showCompiler ( )
```

Definition at line 5 of file compiler.cpp.

Referenced by _RInside_showCompiler().

7.76.2 Variable Documentation

7.76.2.1 CallEntries

```
const R_CallMethodDef CallEntries[]  [static]
```

Initial value:

```
= {
    {"_RInside_showCompiler", (DL_FUNC) &_RInside_showCompiler, 0},
    {NULL, NULL, 0}
}
```

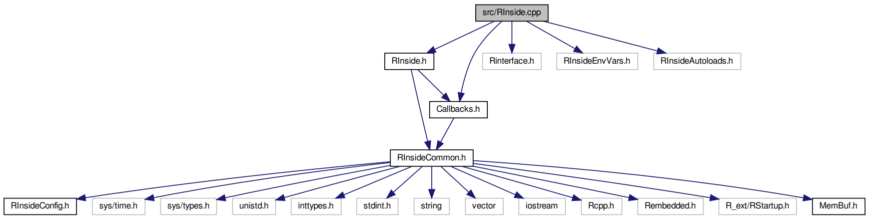
Definition at line 19 of file RcppExports.cpp.

Referenced by R_init_RInside().

7.77 src/RInside.cpp File Reference

```
#include <RInside.h>
#include <Callbacks.h>
#include <Rinterface.h>
#include "RInsideEnvVars.h"
#include "RInsideAutoloads.h"
```

Include dependency graph for RInside.cpp:



Macros

- #define R_INTERFACE_PTRS

Variables

- const char ∗ programName = "RInside"

7.77.1 Macro Definition Documentation

7.77.1.1 R_INTERFACE_PTRS

```
#define R_INTERFACE_PTRS
```

Definition at line 26 of file RInside.cpp.

7.77.2 Variable Documentation

7.77.2.1 programName

```
const char* programName = "RInside"
```

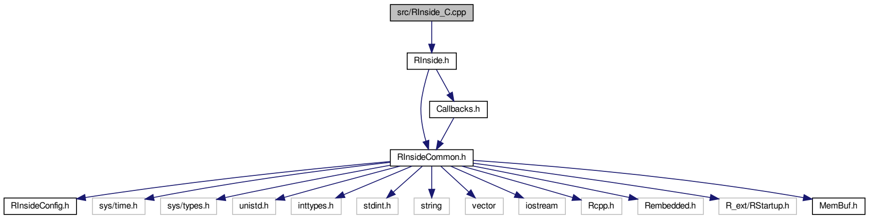
Definition at line 32 of file RInside.cpp.

Referenced by RInside::initialize(), and RInside::parseEval().

7.78 src/RInside_C.cpp File Reference

```
#include <RInside.h>
```

Include dependency graph for RInside_C.cpp:



Functions

- void setupRinC ()
- void passToR (SEXP x, char ∗name)
- SEXP evalInR (char ∗cmd)
- void evalQuietlyInR (char ∗cmd)
- void teardownRinC ()

Variables

- RInside ∗ rr = NULL

7.78.1 Function Documentation

7.78.1.1 evalInR()

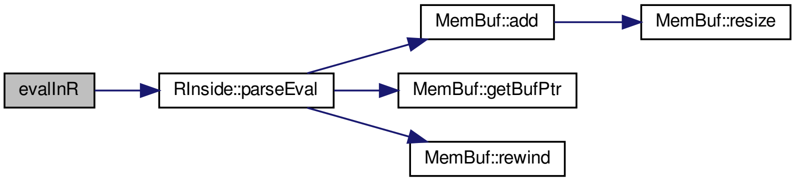
```
SEXP evalInR (
             char * cmd )
```

Definition at line 36 of file RInside_C.cpp.

References RInside::parseEval().

Referenced by main().

Here is the call graph for this function:



7.78.1.2 evalQuietlyInR()

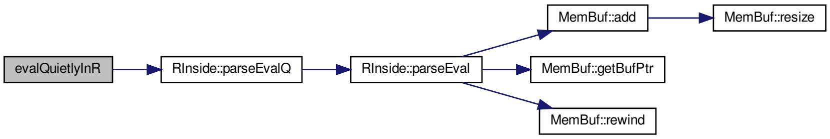
```
void evalQuietlyInR (
             char * cmd )
```

Definition at line 43 of file RInside_C.cpp.

References RInside::parseEvalQ().

Referenced by main().

Here is the call graph for this function:



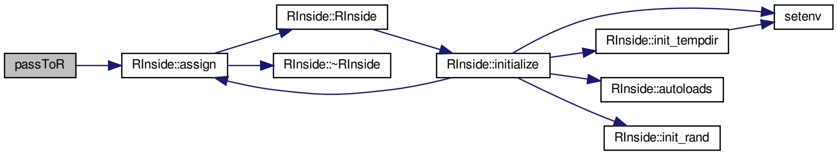
7.78.1.3 passToR()

```
void passToR (
             SEXP x,
             char * name )
```

Definition at line 31 of file RInside_C.cpp.

References RInside::assign().

Here is the call graph for this function:



7.78.1.4 setupRinC()

```
void setupRinC ( )
```

Definition at line 26 of file RInside_C.cpp.

Referenced by main().

7.78.1.5 teardownRinC()

```
void teardownRinC ( )
```

Definition at line 48 of file RInside_C.cpp.

References rr.

Referenced by main().

7.78.2 Variable Documentation

7.78.2.1 rr

```
RInside* rr = NULL
```

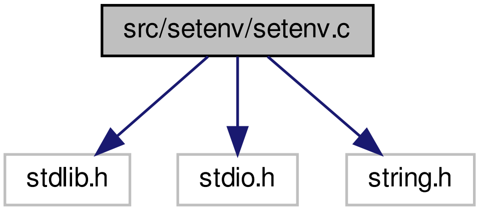
Definition at line 23 of file RInside_C.cpp.

Referenced by teardownRinC().

7.79 src/setenv/setenv.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
```

Include dependency graph for setenv.c:



Functions

- int setenv (const char ∗env_var, const char ∗env_val, int dummy)

7.79.1 Function Documentation

7.79.1.1 setenv()

```
int setenv (
             const char * env_var,
             const char * env_val,
             int dummy )
```

Definition at line 28 of file setenv.c.

Referenced by RInside::init_tempdir(), and RInside::initialize().

7.80 src/tools/RInsideAutoloads.r File Reference

7.81 src/tools/RInsideEnvVars.r File Reference

7.82 src/tools/unix2dos.r File Reference