EquityOpt

Generated by Doxygen 1.8.11

Contents

Chapter 1

Test List

Class QuantLib::Bond (p. **45)**

price/yield calculations are cross-checked for consistency.

- price/yield calculations are checked against known good values.

Class QuantLib::CapFloor (p. **53)**

the correctness of the returned value is tested by checking that the price of a cap (resp. floor) decreases (resp. increases) with the strike rate.

- the relationship between the values of caps, floors and the resulting collars is checked.
- the put-call parity between the values of caps, floors and swaps is checked.
- the correctness of the returned implied volatility is tested by using it for reproducing the target value.
- the correctness of the returned value is tested by checking it against a known good value.

Class QuantLib::CmsRateBond (p. **58)**

calculations are tested by checking results against cached values.

Class QuantLib::FixedRateBond (p. **87)**

calculations are tested by checking results against cached values.

Class QuantLib::FloatingRateBond (p. **93)**

calculations are tested by checking results against cached values.

Class QuantLib::Swaption (p. **138)**

the correctness of the returned value is tested by checking that the price of a payer (resp. receiver) swaption decreases (resp. increases) with the strike.

- the correctness of the returned value is tested by checking that the price of a payer (resp. receiver) swaption increases (resp. decreases) with the spread.
- the correctness of the returned value is tested by checking it against that of a swaption on a swap with no spread and a correspondingly adjusted fixed rate.
- the correctness of the returned value is tested by checking it against a known good value.
- the correctness of the returned value of cash settled swaptions is tested by checking the modified annuity against a value calculated without using the **Swaption** (p. 138) class.

Class QuantLib::VanillaSwap (p. **142)**

the correctness of the returned value is tested by checking that the price of a swap paying the fair fixed rate is null.

- the correctness of the returned value is tested by checking that the price of a swap receiving the fair floating-rate spread is null.
- the correctness of the returned value is tested by checking that the price of a swap decreases with the paid fixed rate.

- the correctness of the returned value is tested by checking that the price of a swap increases with the received floating-rate spread.
- the correctness of the returned value is tested by checking it against a known good value.

Class QuantLib::YoYInflationCapFloor (p. **148**)

the relationship between the values of caps, floors and the resulting collars is checked.

- the put-call parity between the values of caps, floors and swaps is checked.
- the correctness of the returned value is tested by checking it against a known good value.

Class QuantLib::ZeroCouponBond (p. **150**)

calculations are tested by checking results against cached values.

Chapter 2

Todo List

Class QuantLib::CliquetOption (p. **57)**

add local/global caps/floors

- add accrued coupon and last fixing

Class QuantLib::ContinuousAveragingAsianOption (p. **60)**

add running average

Class QuantLib::FixedRateBondForward (p. **88)**

Add preconditions and tests

Create switch- if coupon goes to seller is toggled on, don't consider income in the $P_{DirtyFwd}(t)$ calculation.

Verify this works when the underlying is paper (in which case ignore all AI.)

Class QuantLib::Forward (p. **95)**

Add preconditions and tests

Class QuantLib::Swaption (p. **138)**

add greeks and explicit exercise lag

Chapter 3

Bug List

Class QuantLib::AssetSwap (p. **40**)

fair prices are not calculated correctly when using indexed coupons.

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

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

Chapter 5

Class Index

5.1 Class List

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

Chapter 6

File Index

6.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 7

Class Documentation

7.1 QuantLib::DiscreteAveragingAsianOption::arguments Class Reference

Extra arguments for single-asset discrete-average Asian option.

`#include <asianoption.hpp>`

Inheritance diagram for QuantLib::DiscreteAveragingAsianOption::arguments:

Collaboration diagram for QuantLib::DiscreteAveragingAsianOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Average::Type **averageType**
- Real **runningAccumulator**
- Size **pastFixings**
- std::vector< Date > **fixingDates**

7.1.1 Detailed Description

Extra arguments for single-asset discrete-average Asian option.

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

- C:/quantlib/QuantLib/ql/instruments/**asianoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/asianoption.cpp

7.2 QuantLib::ContinuousAveragingAsianOption::arguments Class Reference

Extra arguments for single-asset continuous-average Asian option.

`#include <asianoption.hpp>`

Inheritance diagram for QuantLib::ContinuousAveragingAsianOption::arguments:

Collaboration diagram for QuantLib::ContinuousAveragingAsianOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Average::Type **averageType**

7.2.1 Detailed Description

Extra arguments for single-asset continuous-average Asian option.

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

- C:/quantlib/QuantLib/ql/instruments/**asianoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/asianoption.cpp

7.3 QuantLib::AssetSwap::arguments Class Reference

Arguments for asset swap calculation

`#include <assetswap.hpp>`

Inheritance diagram for QuantLib::AssetSwap::arguments:

Collaboration diagram for QuantLib::AssetSwap::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- std::vector< Date > **fixedResetDates**
- std::vector< Date > **fixedPayDates**
- std::vector< Real > **fixedCoupons**
- std::vector< Time > **floatingAccrualTimes**
- std::vector< Date > **floatingResetDates**
- std::vector< Date > **floatingFixingDates**
- std::vector< Date > **floatingPayDates**
- std::vector< Spread > **floatingSpreads**

7.3.1 Detailed Description

Arguments for asset swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**assetswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/assetswap.cpp

7.4 QuantLib::CreditDefaultSwap::arguments Class Reference

Inheritance diagram for QuantLib::CreditDefaultSwap::arguments:

Collaboration diagram for QuantLib::CreditDefaultSwap::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Protection::Side **side**
- Real **notional**
- boost::optional< Rate > **upfront**
- Rate **spread**
- Leg **leg**
- boost::shared_ptr< CashFlow > **upfrontPayment**
- bool **settlesAccrual**
- bool **paysAtDefaultTime**
- boost::shared_ptr< **Claim** > **claim**
- Date **protectionStart**

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

- C:/quantlib/QuantLib/ql/instruments/**creditdefaultswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/creditdefaultswap.cpp

7.5 QuantLib::DividendBarrierOption::arguments Class Reference

Arguments for dividend barrier option calculation

`#include <dividendbarrieroption.hpp>`

Inheritance diagram for QuantLib::DividendBarrierOption::arguments:

Collaboration diagram for QuantLib::DividendBarrierOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- DividendSchedule **cashFlow**

7.5.1 Detailed Description

Arguments for dividend barrier option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**dividendbarrieroption.hpp**
- C:/quantlib/QuantLib/ql/instruments/dividendbarrieroption.cpp

7.6 QuantLib::DividendVanillaOption::arguments Class Reference

Arguments for dividend vanilla option calculation

`#include <dividendvanillaoption.hpp>`

Inheritance diagram for QuantLib::DividendVanillaOption::arguments:

Collaboration diagram for QuantLib::DividendVanillaOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- DividendSchedule **cashFlow**

7.6.1 Detailed Description

Arguments for dividend vanilla option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**dividendvanillaoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/dividendvanillaoption.cpp

7.7 QuantLib::FloatFloatSwap::arguments Class Reference

Arguments for float float swap calculation

`#include <floatfloatswap.hpp>`

Inheritance diagram for QuantLib::FloatFloatSwap::arguments:

Collaboration diagram for QuantLib::FloatFloatSwap::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- VanillaSwap::Type **type**
- std::vector< Real > **nominal1**
- std::vector< Real > **nominal2**
- std::vector< Date > **leg1ResetDates**
- std::vector< Date > **leg1FixingDates**
- std::vector< Date > **leg1PayDates**
- std::vector< Date > **leg2ResetDates**
- std::vector< Date > **leg2FixingDates**
- std::vector< Date > **leg2PayDates**
- std::vector< Real > **leg1Spreads**
- std::vector< Real > **leg2Spreads**
- std::vector< Real > **leg1Gearings**
- std::vector< Real > **leg2Gearings**
- std::vector< Real > **leg1CappedRates**
- std::vector< Real > **leg1FlooredRates**
- std::vector< Real > **leg2CappedRates**
- std::vector< Real > **leg2FlooredRates**
- std::vector< Real > **leg1Coupons**
- std::vector< Real > **leg2Coupons**
- std::vector< Real > **leg1AccrualTimes**
- std::vector< Real > **leg2AccrualTimes**
- boost::shared_ptr< InterestRateIndex > **index1**
- boost::shared_ptr< InterestRateIndex > **index2**
- std::vector< bool > **leg1IsRedemptionFlow**
- std::vector< bool > **leg2IsRedemptionFlow**

7.7.1 Detailed Description

Arguments for float float swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**floatfloatswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/floatfloatswap.cpp

7.8 QuantLib::Swap::arguments Class Reference

Inheritance diagram for QuantLib::Swap::arguments:

Collaboration diagram for QuantLib::Swap::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- std::vector< Leg > **legs**
- std::vector< Real > **payer**

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

- C:/quantlib/QuantLib/ql/instruments/**swap.hpp**
- C:/quantlib/QuantLib/ql/instruments/swap.cpp

7.9 QuantLib::Swaption::arguments Class Reference

Arguments for swaption calculation

`#include <swaption.hpp>`

Inheritance diagram for QuantLib::Swaption::arguments:

Collaboration diagram for QuantLib::Swaption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- boost::shared_ptr< **VanillaSwap** > **swap**
- Settlement::Type **settlementType**

7.9.1 Detailed Description

Arguments for swaption calculation

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

- C:/quantlib/QuantLib/ql/instruments/**swaption.hpp**
- C:/quantlib/QuantLib/ql/instruments/swaption.cpp

7.10 QuantLib::CapFloor::arguments Class Reference

Arguments for cap/floor calculation

`#include <capfloor.hpp>`

Inheritance diagram for QuantLib::CapFloor::arguments:

Collaboration diagram for QuantLib::CapFloor::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- CapFloor::Type **type**
- std::vector< Date > **startDates**
- std::vector< Date > **fixingDates**
- std::vector< Date > **endDates**
- std::vector< Time > **accrualTimes**
- std::vector< Rate > **capRates**
- std::vector< Rate > **floorRates**
- std::vector< Rate > **forwards**
- std::vector< Real > **gearings**
- std::vector< Real > **spreads**
- std::vector< Real > **nominals**
- std::vector< boost::shared_ptr< InterestRateIndex > > **indexes**

7.10.1 Detailed Description

Arguments for cap/floor calculation

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

- C:/quantlib/QuantLib/ql/instruments/**capfloor.hpp**
- C:/quantlib/QuantLib/ql/instruments/capfloor.cpp

7.11 QuantLib::FloatFloatSwaption::arguments Class Reference

Arguments for cms swaption calculation

`#include <floatfloatswaption.hpp>`

Inheritance diagram for QuantLib::FloatFloatSwaption::arguments:

Collaboration diagram for QuantLib::FloatFloatSwaption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- boost::shared_ptr< **FloatFloatSwap** > **swap**

7.11.1 Detailed Description

Arguments for cms swaption calculation

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

- C:/quantlib/QuantLib/ql/instruments/**floatfloatswaption.hpp**
- C:/quantlib/QuantLib/ql/instruments/floatfloatswaption.cpp

7.12 QuantLib::VanillaStorageOption::arguments Class Reference

Inheritance diagram for QuantLib::VanillaStorageOption::arguments:

Collaboration diagram for QuantLib::VanillaStorageOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Real **capacity**
- Real **load**
- Real **changeRate**
- boost::shared_ptr< **NullPayoff** > **payoff**
- boost::shared_ptr< BermudanExercise > **exercise**

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

- C:/quantlib/QuantLib/ql/instruments/**vanillastorageoption.hpp**

7.13 QuantLib::VanillaSwap::arguments Class Reference

Arguments for simple swap calculation

`#include <vanillaswap.hpp>`

Inheritance diagram for QuantLib::VanillaSwap::arguments:

Collaboration diagram for QuantLib::VanillaSwap::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Type **type**
- Real **nominal**
- std::vector< Date > **fixedResetDates**
- std::vector< Date > **fixedPayDates**
- std::vector< Time > **floatingAccrualTimes**
- std::vector< Date > **floatingResetDates**
- std::vector< Date > **floatingFixingDates**
- std::vector< Date > **floatingPayDates**
- std::vector< Real > **fixedCoupons**
- std::vector< Spread > **floatingSpreads**
- std::vector< Real > **floatingCoupons**

7.13.1 Detailed Description

Arguments for simple swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**vanillaswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/vanillaswap.cpp

7.14 QuantLib::VanillaSwingOption::arguments Class Reference

Inheritance diagram for QuantLib::VanillaSwingOption::arguments:

Collaboration diagram for QuantLib::VanillaSwingOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Size **minExerciseRights**
- Size **maxExerciseRights**
- boost::shared_ptr< **StrikedTypePayoff** > **payoff**
- boost::shared_ptr< **SwingExercise** > **exercise**

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

- C:/quantlib/QuantLib/ql/instruments/**vanillaswingoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/**vanillaswingoption.cpp**

7.15 QuantLib::VarianceSwap::arguments Class Reference

Arguments for forward fair-variance calculation

`#include <varianceswap.hpp>`

Inheritance diagram for QuantLib::VarianceSwap::arguments:

Collaboration diagram for QuantLib::VarianceSwap::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Position::Type **position**
- Real **strike**
- Real **notional**
- Date **startDate**
- Date **maturityDate**

7.15.1 Detailed Description

Arguments for forward fair-variance calculation

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

- C:/quantlib/QuantLib/ql/instruments/**varianceswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/varianceswap.cpp

7.16 QuantLib::YearOnYearInflationSwap::arguments Class Reference

Arguments for YoY swap calculation

`#include <yearonyearinflationswap.hpp>`

Inheritance diagram for QuantLib::YearOnYearInflationSwap::arguments:

Collaboration diagram for QuantLib::YearOnYearInflationSwap::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Type **type**
- Real **nominal**
- std::vector< Date > **fixedResetDates**
- std::vector< Date > **fixedPayDates**
- std::vector< Time > **yoyAccrualTimes**
- std::vector< Date > **yoyResetDates**
- std::vector< Date > **yoyFixingDates**
- std::vector< Date > **yoyPayDates**
- std::vector< Real > **fixedCoupons**
- std::vector< Spread > **yoySpreads**
- std::vector< Real > **yoyCoupons**

7.16.1 Detailed Description

Arguments for YoY swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**yearonyearinflationswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/yearonyearinflationswap.cpp

7.17 QuantLib::YoYInflationCapFloor::arguments Class Reference

Arguments for YoY Inflation cap/floor calculation

`#include <inflationcapfloor.hpp>`

Inheritance diagram for QuantLib::YoYInflationCapFloor::arguments:

Collaboration diagram for QuantLib::YoYInflationCapFloor::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- YoYInflationCapFloor::Type **type**
- boost::shared_ptr< YoYInflationIndex > **index**
- Period **observationLag**
- std::vector< Date > **startDates**
- std::vector< Date > **fixingDates**
- std::vector< Date > **payDates**
- std::vector< Time > **accrualTimes**
- std::vector< Rate > **capRates**
- std::vector< Rate > **floorRates**
- std::vector< Real > **gearings**
- std::vector< Real > **spreads**
- std::vector< Real > **nominals**

7.17.1 Detailed Description

Arguments for YoY Inflation cap/floor calculation

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

- C:/quantlib/QuantLib/ql/instruments/inflationcapfloor.hpp
- C:/quantlib/QuantLib/ql/instruments/inflationcapfloor.cpp

7.18 QuantLib::ZeroCouponInflationSwap::arguments Class Reference

Inheritance diagram for QuantLib::ZeroCouponInflationSwap::arguments:

Collaboration diagram for QuantLib::ZeroCouponInflationSwap::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Rate **fixedRate**

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

- C:/quantlib/QuantLib/ql/instruments/**zerocouponinflationswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/zerocouponinflationswap.cpp

7.19 QuantLib::BarrierOption::arguments Class Reference

Arguments for barrier option calculation

```
#include <barrieroption.hpp>
```

Inheritance diagram for QuantLib::BarrierOption::arguments:

Collaboration diagram for QuantLib::BarrierOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Barrier::Type **barrierType**
- Real **barrier**
- Real **rebate**

7.19.1 Detailed Description

Arguments for barrier option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**barrieroption.hpp**
- C:/quantlib/QuantLib/ql/instruments/barrieroption.cpp

7.20 QuantLib::Bond::arguments Class Reference

Inheritance diagram for QuantLib::Bond::arguments:

Collaboration diagram for QuantLib::Bond::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Date **settlementDate**
- Leg **cashflows**
- Calendar **calendar**

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

- C:/quantlib/QuantLib/ql/instruments/**bond.hpp**
- C:/quantlib/QuantLib/ql/instruments/bond.cpp

7.21 QuantLib::CliquetOption::arguments Class Reference

Arguments for cliquet option calculation

```
#include <cliquetoption.hpp>
```

Inheritance diagram for QuantLib::CliquetOption::arguments:

Collaboration diagram for QuantLib::CliquetOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Real **accruedCoupon**
- Real **lastFixing**
- Real **localCap**
- Real **localFloor**
- Real **globalCap**
- Real **globalFloor**
- std::vector< Date > **resetDates**

7.21.1 Detailed Description

Arguments for cliquet option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**cliquetoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/cliquetoption.cpp

7.22 QuantLib::ContinuousFloatingLookbackOption::arguments Class Reference

Arguments for continuous floating lookback option calculation

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousFloatingLookbackOption::arguments:

Collaboration diagram for QuantLib::ContinuousFloatingLookbackOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Real **minmax**

7.22.1 Detailed Description

Arguments for continuous floating lookback option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/lookbackoption.cpp

7.23 QuantLib::ContinuousFixedLookbackOption::arguments Class Reference

Arguments for continuous fixed lookback option calculation

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousFixedLookbackOption::arguments:

Collaboration diagram for QuantLib::ContinuousFixedLookbackOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Real **minmax**

7.23.1 Detailed Description

Arguments for continuous fixed lookback option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/lookbackoption.cpp

7.24 QuantLib::ContinuousPartialFloatingLookbackOption::arguments Class Reference

Arguments for continuous partial floating lookback option calculation

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousPartialFloatingLookbackOption::arguments:

Collaboration diagram for QuantLib::ContinuousPartialFloatingLookbackOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Real **lambda**
- Date **lookbackPeriodEnd**

7.24.1 Detailed Description

Arguments for continuous partial floating lookback option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/lookbackoption.cpp

7.25 QuantLib::ContinuousPartialFixedLookbackOption::arguments Class Reference

Arguments for continuous partial fixed lookback option calculation

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousPartialFixedLookbackOption::arguments:

Collaboration diagram for QuantLib::ContinuousPartialFixedLookbackOption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Date **lookbackPeriodStart**

7.25.1 Detailed Description

Arguments for continuous partial fixed lookback option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/lookbackoption.cpp

7.26 QuantLib::CPICapFloor::arguments Class Reference

Inheritance diagram for QuantLib::CPICapFloor::arguments:

Collaboration diagram for QuantLib::CPICapFloor::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Option::Type **type**
- Real **nominal**
- Date **startDate**
- Date **fixDate**
- Date **payDate**
- Real **baseCPI**
- Date **maturity**
- Calendar **fixCalendar**
- Calendar **payCalendar**
- BusinessDayConvention **fixConvention**
- BusinessDayConvention **payConvention**
- Rate **strike**
- Handle< ZeroInflationIndex > **inflIndex**
- Period **observationLag**
- CPI::InterpolationType **observationInterpolation**

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

- C:/quantlib/QuantLib/ql/instruments/**cpicapfloor.hpp**
- C:/quantlib/QuantLib/ql/instruments/cpicapfloor.cpp

7.27 QuantLib::CPISwap::arguments Class Reference

Arguments for swap calculation

`#include <cpiswap.hpp>`

Inheritance diagram for QuantLib::CPISwap::arguments:

Collaboration diagram for QuantLib::CPISwap::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- Type **type**
- Real **nominal**

7.27.1 Detailed Description

Arguments for swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**cpiswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/cpiswap.cpp

7.28 QuantLib::NonstandardSwap::arguments Class Reference

Arguments for nonstandard swap calculation

`#include <nonstandardswap.hpp>`

Inheritance diagram for QuantLib::NonstandardSwap::arguments:

Collaboration diagram for QuantLib::NonstandardSwap::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- VanillaSwap::Type **type**
- std::vector< Real > **fixedNominal**
- std::vector< Real > **floatingNominal**
- std::vector< Date > **fixedResetDates**
- std::vector< Date > **fixedPayDates**
- std::vector< Time > **floatingAccrualTimes**
- std::vector< Date > **floatingResetDates**
- std::vector< Date > **floatingFixingDates**
- std::vector< Date > **floatingPayDates**
- std::vector< Real > **fixedCoupons**
- std::vector< Real > **fixedRate**
- std::vector< Spread > **floatingSpreads**
- std::vector< Real > **floatingGearings**
- std::vector< Real > **floatingCoupons**
- boost::shared_ptr< IborIndex > **iborIndex**
- std::vector< bool > **fixedIsRedemptionFlow**
- std::vector< bool > **floatingIsRedemptionFlow**

7.28.1 Detailed Description

Arguments for nonstandard swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**nonstandardswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/nonstandardswap.cpp

7.29 QuantLib::NonstandardSwaption::arguments Class Reference

Arguments for nonstandard swaption calculation

`#include <nonstandardswaption.hpp>`

Inheritance diagram for QuantLib::NonstandardSwaption::arguments:

Collaboration diagram for QuantLib::NonstandardSwaption::arguments:

Public Member Functions

- void **validate** () const

Public Attributes

- boost::shared_ptr< **NonstandardSwap** > **swap**
- Settlement::Type **settlementType**

7.29.1 Detailed Description

Arguments for nonstandard swaption calculation

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

- C:/quantlib/QuantLib/ql/instruments/**nonstandardswaption.hpp**
- C:/quantlib/QuantLib/ql/instruments/nonstandardswaption.cpp

7.30 QuantLib::AssetOrNothingPayoff Class Reference

Binary asset-or-nothing payoff.

`#include <payoffs.hpp>`

Inheritance diagram for QuantLib::AssetOrNothingPayoff:

Collaboration diagram for QuantLib::AssetOrNothingPayoff:

Public Member Functions

- **AssetOrNothingPayoff** (Option::Type type, Real strike)

Payoff interface

- std::string **name** () const
- Real **operator()** (Real price) const
- virtual void **accept** (AcyclicVisitor &)

Additional Inherited Members

7.30.1 Detailed Description

Binary asset-or-nothing payoff.

Definitions of Binary path-independent payoffs used below, can be found in M. Rubinstein, E. Reiner:"Unscrambling The Binary Code", Risk, Vol.4 no.9,1991. (see: `http://www.in-the-money.com/artandpap/↩ Binary%20Options.doc`)

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

- C:/quantlib/QuantLib/ql/instruments/**payoffs.hpp**
- C:/quantlib/QuantLib/ql/instruments/payoffs.cpp

7.31 QuantLib::AssetSwap Class Reference

Bullet bond vs Libor swap.

`#include <assetswap.hpp>`

Inheritance diagram for QuantLib::AssetSwap:

Collaboration diagram for QuantLib::AssetSwap:

Classes

- class **arguments**

  Arguments for asset swap calculation
- class **results**

  Results from simple swap calculation

Public Member Functions

- **AssetSwap** (bool payBondCoupon, const boost::shared_ptr< **Bond** > &bond, Real bondCleanPrice, const boost::shared_ptr< IborIndex > &iborIndex, Spread spread, const Schedule &floatSchedule=Schedule(), const DayCounter &floatingDayCount=DayCounter(), bool parAssetSwap=true)
- **AssetSwap** (bool parAssetSwap, const boost::shared_ptr< **Bond** > &bond, Real bondCleanPrice, Real nonParRepayment, Real gearing, const boost::shared_ptr< IborIndex > &iborIndex, Spread spread=0.0, const DayCounter &floatingDayCount=DayCounter(), Date dealMaturity=Date(), bool payBondCoupon=false)
- Spread **fairSpread** () const
- Real **floatingLegBPS** () const
- Real **floatingLegNPV** () const
- Real **fairCleanPrice** () const
- Real **fairNonParRepayment** () const
- bool **parSwap** () const
- Spread **spread** () const
- Real **cleanPrice** () const
- Real **nonParRepayment** () const
- const boost::shared_ptr< **Bond** > & **bond** () const
- bool **payBondCoupon** () const
- const Leg & **bondLeg** () const
- const Leg & **floatingLeg** () const
- void **setupArguments** (PricingEngine::arguments *args) const
- void **fetchResults** (const PricingEngine::results *) const

Additional Inherited Members

7.31.1 Detailed Description

Bullet bond vs Libor swap.

for mechanics of par asset swap and market asset swap, refer to "Introduction to Asset Swap", Lehman Brothers European Fixed Income Research - January 2000, D. O'Kane

Warning

bondCleanPrice must be the (forward) price at the floatSchedule start date

Bug fair prices are not calculated correctly when using indexed coupons.

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

- C:/quantlib/QuantLib/ql/instruments/**assetswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/assetswap.cpp

7.32 QuantLib::Average Struct Reference

Placeholder for enumerated averaging types.

`#include <averagetype.hpp>`

Public Types

- enum **Type** { **Arithmetic**, **Geometric** }

7.32.1 Detailed Description

Placeholder for enumerated averaging types.

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

- C:/quantlib/QuantLib/ql/instruments/**averagetype.hpp**

7.33 QuantLib::AverageBasketPayoff Class Reference

Inheritance diagram for QuantLib::AverageBasketPayoff:

Collaboration diagram for QuantLib::AverageBasketPayoff:

Public Member Functions

- **AverageBasketPayoff** (const boost::shared_ptr< Payoff > &p, const Array &a)
- **AverageBasketPayoff** (const boost::shared_ptr< Payoff > &p, Size n)
- Real **accumulate** (const Array &a) const

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

- C:/quantlib/QuantLib/ql/instruments/**basketoption.hpp**

7.34 QuantLib::Barrier Struct Reference

Placeholder for enumerated barrier types.

`#include <barriertype.hpp>`

Public Types

- enum **Type** { **DownIn**, **UpIn**, **DownOut**, **UpOut** }

7.34.1 Detailed Description

Placeholder for enumerated barrier types.

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

- C:/quantlib/QuantLib/ql/instruments/**barriertype.hpp**

7.35 QuantLib::BarrierOption Class Reference

Barrier option on a single asset.

`#include <barrieroption.hpp>`

Inheritance diagram for QuantLib::BarrierOption:

Collaboration diagram for QuantLib::BarrierOption:

Classes

- class **arguments**
  Arguments for barrier option calculation
- class **engine**
  Barrier-option engine base class

Public Member Functions

- **BarrierOption** (Barrier::Type barrierType, Real barrier, Real rebate, const boost::shared_ptr< **StrikedType**↩ **Payoff** > &payoff, const boost::shared_ptr< Exercise > &exercise)
- void **setupArguments** (PricingEngine::arguments *) const
- Volatility **impliedVolatility** (Real price, const boost::shared_ptr< GeneralizedBlackScholesProcess > &process, Real accuracy=1.0e-4, Size maxEvaluations=100, Volatility minVol=1.0e-7, Volatility maxVol=4.0) const

Protected Attributes

- Barrier::Type **barrierType_**
- Real **barrier_**
- Real **rebate_**

Additional Inherited Members

7.35.1 Detailed Description

Barrier option on a single asset.

The analytic pricing engine will be used if none if passed.

7.35.2 Member Function Documentation

7.35.2.1 Volatility QuantLib::BarrierOption::impliedVolatility (Real *price*, const boost::shared_ptr< GeneralizedBlackScholesProcess > & *process*, Real *accuracy* = 1.0e-4, Size *maxEvaluations* = 100, Volatility *minVol* = 1.0e-7, Volatility *maxVol* = 4.0) const

Warning

see **VanillaOption** (p. 140) for notes on implied-volatility calculation.

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

- C:/quantlib/QuantLib/ql/instruments/**barrieroption.hpp**
- C:/quantlib/QuantLib/ql/instruments/barrieroption.cpp

7.36 QuantLib::BasketOption Class Reference

Basket option on a number of assets.

`#include <basketoption.hpp>`

Inheritance diagram for QuantLib::BasketOption:

Collaboration diagram for QuantLib::BasketOption:

Classes

- class **engine**
  Basket-option engine base class

Public Member Functions

- **BasketOption** (const boost::shared_ptr< **BasketPayoff** > &, const boost::shared_ptr< Exercise > &)

Additional Inherited Members

7.36.1 Detailed Description

Basket option on a number of assets.

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

- C:/quantlib/QuantLib/ql/instruments/**basketoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/basketoption.cpp

7.37 QuantLib::BasketPayoff Class Reference

Inheritance diagram for QuantLib::BasketPayoff:

Collaboration diagram for QuantLib::BasketPayoff:

Public Member Functions

- **BasketPayoff** (const boost::shared_ptr< Payoff > &p)
- std::string **name** () const
- std::string **description** () const
- Real **operator()** (Real price) const
- virtual Real **operator()** (const Array &a) const
- virtual Real **accumulate** (const Array &a) const =0
- const boost::shared_ptr< Payoff > **basePayoff** ()

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

- C:/quantlib/QuantLib/ql/instruments/**basketoption.hpp**

7.38 QuantLib::BMASwap Class Reference

swap paying Libor against BMA coupons

`#include <bmaswap.hpp>`

Inheritance diagram for QuantLib::BMASwap:

Collaboration diagram for QuantLib::BMASwap:

Public Types

- enum **Type** { **Receiver** = -1, **Payer** = 1 }

Public Member Functions

- **BMASwap** (Type **type**, Real nominal, const Schedule &liborSchedule, Rate liborFraction, Rate libor↩ Spread, const boost::shared_ptr< IborIndex > &liborIndex, const DayCounter &liborDayCount, const Schedule &bmaSchedule, const boost::shared_ptr< BMAIndex > &bmaIndex, const DayCounter &bmaDayCount)

Inspectors

- Real **liborFraction** () const
- Spread **liborSpread** () const
- Real **nominal** () const
- Type **type** () const
  "payer" or "receiver" refer to the BMA leg
- const Leg & **bmaLeg** () const
- const Leg & **liborLeg** () const

Results

- Real **liborLegBPS** () const
- Real **liborLegNPV** () const
- Rate **fairLiborFraction** () const
- Spread **fairLiborSpread** () const
- Real **bmaLegBPS** () const
- Real **bmaLegNPV** () const

Additional Inherited Members

7.38.1 Detailed Description

swap paying Libor against BMA coupons

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

- C:/quantlib/QuantLib/ql/instruments/**bmaswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/bmaswap.cpp

7.39 QuantLib::Bond Class Reference

Base bond class.

`#include <bond.hpp>`

Inheritance diagram for QuantLib::Bond:

Collaboration diagram for QuantLib::Bond:

Classes

- class **arguments**
- class **engine**
- class **results**

Public Member Functions

- **Bond** (Natural settlementDays, const Calendar &calendar, const Date &issueDate=Date(), const Leg &coupons=Leg())

  constructor for amortizing or non-amortizing bonds.
- **Bond** (Natural settlementDays, const Calendar &calendar, Real faceAmount, const Date &maturityDate, const Date &issueDate=Date(), const Leg &**cashflows**=Leg())

  old constructor for non amortizing bonds.
- virtual Rate **nextCouponRate** (Date d=Date()) const
- Rate **previousCouponRate** (Date d=Date()) const

  Previous coupon already paid at a given date.
- Date **nextCashFlowDate** (Date d=Date()) const
- Date **previousCashFlowDate** (Date d=Date()) const

Instrument interface

- bool **isExpired** () const

Inspectors

- Natural **settlementDays** () const
- const Calendar & **calendar** () const

- const std::vector< Real > & **notionals** () const
- virtual Real **notional** (Date d=Date()) const
- const Leg & **cashflows** () const
- const Leg & **redemptions** () const
- const boost::shared_ptr< CashFlow > & **redemption** () const
- Date **startDate** () const
- Date **maturityDate** () const
- Date **issueDate** () const
- bool **isTradable** (Date d=Date()) const
- Date **settlementDate** (Date d=Date()) const

Calculations

- Real **cleanPrice** () const
  theoretical clean price
- Real **dirtyPrice** () const
  theoretical dirty price
- Real **settlementValue** () const
  theoretical settlement value
- Rate **yield** (const DayCounter &dc, Compounding comp, Frequency freq, Real accuracy=1.0e-8, Size maxEvaluations=100) const
  theoretical bond yield
- Real **cleanPrice** (Rate **yield**, const DayCounter &dc, Compounding comp, Frequency freq, Date settlementDate=Date()) const
  clean price given a yield and settlement date
- Real **dirtyPrice** (Rate **yield**, const DayCounter &dc, Compounding comp, Frequency freq, Date settlementDate=Date()) const
  dirty price given a yield and settlement date
- Real **settlementValue** (Real **cleanPrice**) const
  settlement value as a function of the clean price
- Rate **yield** (Real **cleanPrice**, const DayCounter &dc, Compounding comp, Frequency freq, Date settlementDate=Date(), Real accuracy=1.0e-8, Size maxEvaluations=100) const
  yield given a (clean) price and settlement date
- virtual Real **accruedAmount** (Date d=Date()) const
  accrued amount at a given date

Protected Member Functions

- void **setupExpired** () const
- void **setupArguments** (PricingEngine::arguments *) const
- void **fetchResults** (const PricingEngine::results *) const
- void **addRedemptionsToCashflows** (const std::vector< Real > &**redemptions**=std::vector< Real >())
- void **setSingleRedemption** (Real notional, Real **redemption**, const Date &date)
- void **setSingleRedemption** (Real notional, const boost::shared_ptr< CashFlow > &**redemption**)
- void **calculateNotionalsFromCashflows** ()

Protected Attributes

- Natural **settlementDays_**
- Calendar **calendar_**
- std::vector< Date > **notionalSchedule_**
- std::vector< Real > **notionals_**
- Leg **cashflows_**
- Leg **redemptions_**
- Date **maturityDate_**
- Date **issueDate_**
- Real **settlementValue_**

7.39.1 Detailed Description

Base bond class.

Derived classes must fill the uninitialized data members.

Warning

Most methods assume that the cash flows are stored sorted by date, the redemption(s) being after any cash flow at the same date. In particular, if there's one single redemption, it must be the last cash flow,

Test
- price/yield calculations are cross-checked for consistency.
- price/yield calculations are checked against known good values.

7.39.2 Constructor & Destructor Documentation

7.39.2.1 QuantLib::Bond::Bond (Natural *settlementDays,* const Calendar & *calendar,* const Date & *issueDate* = `Date()`, const Leg & *coupons* = `Leg()`)

constructor for amortizing or non-amortizing bonds.

Redemptions and maturity are calculated from the coupon data, if available. Therefore, redemptions must not be included in the passed cash flows.

7.39.2.2 QuantLib::Bond::Bond (Natural *settlementDays,* const Calendar & *calendar,* Real *faceAmount,* const Date & *maturityDate,* const Date & *issueDate* = `Date()`, const Leg & *cashflows* = `Leg()`)

old constructor for non amortizing bonds.

Warning

The last passed cash flow must be the bond redemption. No other cash flow can have a date later than the redemption date.

7.39.3 Member Function Documentation

7.39.3.1 Real QuantLib::Bond::accruedAmount (Date *d* = `Date()`) const `[virtual]`

accrued amount at a given date

The default bond settlement is used if no date is given.

Reimplemented in **QuantLib::BTP** (p. 51), and **QuantLib::CCTEU** (p. 56).

7.39.3.2 void QuantLib::Bond::addRedemptionsToCashflows (const std::vector< Real > & *redemptions* = std::vector<Real>()) [protected]

This method can be called by derived classes in order to build redemption payments from the existing cash flows. It must be called after setting up the cashflows_ vector and will fill the notionalSchedule_, notionals_, and redemptions_ data members.

If given, the elements of the redemptions vector will multiply the amount of the redemption cash flow. The elements will be taken in base 100, i.e., a redemption equal to 100 does not modify the amount.

Precondition

The cashflows_ vector must contain at least one coupon and must be sorted by date.

7.39.3.3 void QuantLib::Bond::calculateNotionalsFromCashflows ( ) [protected]

used internally to collect notional information from the coupons. It should not be called by derived classes, unless they already provide redemption cash flows (in which case they must set up the redemptions_ data member independently). It will fill the notionalSchedule_ and notionals_ data members.

7.39.3.4 const Leg & QuantLib::Bond::cashflows ( ) const [inline]

Note

returns all the cashflows, including the redemptions.

7.39.3.5 Real QuantLib::Bond::cleanPrice ( ) const

theoretical clean price

The default bond settlement is used for calculation.

Warning

the theoretical price calculated from a flat term structure might differ slightly from the price calculated from the corresponding yield by means of the other overload of this function. If the price from a constant yield is desired, it is advisable to use such other overload.

7.39.3.6 Real QuantLib::Bond::cleanPrice (Rate *yield,* const DayCounter & *dc,* Compounding *comp,* Frequency *freq,* Date *settlementDate* = Date()) const

clean price given a yield and settlement date

The default bond settlement is used if no date is given.

7.39.3.7 Real QuantLib::Bond::dirtyPrice ( ) const

theoretical dirty price

The default bond settlement is used for calculation.

Warning

the theoretical price calculated from a flat term structure might differ slightly from the price calculated from the corresponding yield by means of the other overload of this function. If the price from a constant yield is desired, it is advisable to use such other overload.

7.39.3.8 Real QuantLib::Bond::dirtyPrice (Rate *yield,* const DayCounter & *dc,* Compounding *comp,* Frequency *freq,* Date *settlementDate* = `Date()`) const

dirty price given a yield and settlement date

The default bond settlement is used if no date is given.

7.39.3.9 Rate QuantLib::Bond::nextCouponRate (Date *d* = `Date()`) const [virtual]

Expected next coupon: depending on (the bond and) the given date the coupon can be historic, deterministic or expected in a stochastic sense. When the bond settlement date is used the coupon is the already-fixed not-yet-paid one.

The current bond settlement is used if no date is given.

7.39.3.10 Rate QuantLib::Bond::previousCouponRate (Date *d* = `Date()`) const

Previous coupon already paid at a given date.

Expected previous coupon: depending on (the bond and) the given date the coupon can be historic, deterministic or expected in a stochastic sense. When the bond settlement date is used the coupon is the last paid one.

The current bond settlement is used if no date is given.

7.39.3.11 const shared_ptr< CashFlow > & QuantLib::Bond::redemption ( ) const

returns the redemption, if only one is defined

7.39.3.12 const Leg & QuantLib::Bond::redemptions ( ) const [inline]

returns just the redemption flows (not interest payments)

7.39.3.13 void QuantLib::Bond::setSingleRedemption (Real *notional,* Real *redemption,* const Date & *date*) [protected]

This method can be called by derived classes in order to build a bond with a single redemption payment. It will fill the notionalSchedule_, notionals_, and redemptions_ data members.

7.39.3.14 void QuantLib::Bond::setSingleRedemption (Real *notional,* const boost::shared_ptr< CashFlow > & *redemption*) [protected]

This method can be called by derived classes in order to build a bond with a single redemption payment. It will fill the notionalSchedule_, notionals_, and redemptions_ data members.

7.39.3.15 Real QuantLib::Bond::settlementValue ( ) const

theoretical settlement value

The default bond settlement date is used for calculation.

7.39.3.16 Real QuantLib::Bond::settlementValue (Real *cleanPrice*) const

settlement value as a function of the clean price

The default bond settlement date is used for calculation.

7.39.3.17 Rate QuantLib::Bond::yield (const DayCounter & *dc,* Compounding *comp,* Frequency *freq,* Real *accuracy* = 1.0e-8, Size *maxEvaluations* = 100) const

theoretical bond yield

The default bond settlement and theoretical price are used for calculation.

7.39.3.18 Rate QuantLib::Bond::yield (Real *cleanPrice,* const DayCounter & *dc,* Compounding *comp,* Frequency *freq,* Date *settlementDate* = Date(), Real *accuracy* = 1.0e-8, Size *maxEvaluations* = 100) const

yield given a (clean) price and settlement date

The default bond settlement is used if no date is given.

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

- C:/quantlib/QuantLib/ql/instruments/**bond.hpp**
- C:/quantlib/QuantLib/ql/instruments/bond.cpp

7.40 QuantLib::BTP Class Reference

Italian **BTP** (p. 51) (Buono Poliennali del Tesoro) fixed rate bond.

`#include <btp.hpp>`

Inheritance diagram for QuantLib::BTP:

Collaboration diagram for QuantLib::BTP:

Public Member Functions

- **BTP** (const Date &maturityDate, Rate fixedRate, const Date &startDate=Date(), const Date &issue↩ Date=Date())
- **BTP** (const Date &maturityDate, Rate fixedRate, Real **redemption**, const Date &startDate=Date(), const Date &issueDate=Date())
- Rate **yield** (Real **cleanPrice**, Date settlementDate=Date(), Real accuracy=1.0e-8, Size maxEvaluations=100) const

  BTP *(p. 51) yield given a (clean) price and settlement date.*

Bond interface

- Real **accruedAmount** (Date d=Date()) const

  accrued amount at a given date

Additional Inherited Members

7.40.1 Detailed Description

Italian **BTP** (p. 51) (Buono Poliennali del Tesoro) fixed rate bond.

7.40.2 Constructor & Destructor Documentation

7.40.2.1 QuantLib::BTP::BTP (const Date & *maturityDate,* Rate *fixedRate,* Real *redemption,* const Date & *startDate =* `Date()`, const Date & *issueDate =* `Date()`)

constructor needed for legacy non-par redemption BTPs. As of today the only remaining one is IT123456789012 that will redeem 99.999 on xx-may-2037

7.40.3 Member Function Documentation

7.40.3.1 Real QuantLib::BTP::accruedAmount (Date *d =* `Date()`) const `[inline]`, `[virtual]`

accrued amount at a given date

The default bond settlement is used if no date is given.

Reimplemented from **QuantLib::Bond** (p. 47).

7.40.3.2 Rate QuantLib::BTP::yield (Real *cleanPrice*, Date *settlementDate* = `Date()`, Real *accuracy* = `1.0e-8`, Size *maxEvaluations* = `100`) const

BTP (p. 51) yield given a (clean) price and settlement date.

The default **BTP** (p. 51) conventions are used: Actual/Actual (ISMA), Compounded, Annual. The default bond settlement is used if no date is given.

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

- C:/quantlib/QuantLib/ql/instruments/bonds/**btp.hpp**
- C:/quantlib/QuantLib/ql/instruments/bonds/btp.cpp

7.41 QuantLib::Callability Class Reference

instrument callability

`#include <callabilityschedule.hpp>`

Inheritance diagram for QuantLib::Callability:

Collaboration diagram for QuantLib::Callability:

Classes

- class **Price**
  amount to be paid upon callability

Public Types

- enum **Type** { **Call**, **Put** }
  type of the callability

Public Member Functions

- **Callability** (const **Price** &price, **Type** type, const Date &date)
- const **Price** & **price** () const
- **Type type** () const

Event interface

- Date **date** () const

Visitability

- virtual void **accept** (AcyclicVisitor &)

7.41.1 Detailed Description

instrument callability

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

- C:/quantlib/QuantLib/ql/instruments/**callabilityschedule.hpp**

7.42 QuantLib::Cap Class Reference

Concrete cap class.

`#include <capfloor.hpp>`

Inheritance diagram for QuantLib::Cap:

Collaboration diagram for QuantLib::Cap:

Public Member Functions

- **Cap** (const Leg &floatingLeg, const std::vector< Rate > &exerciseRates)

Additional Inherited Members

7.42.1 Detailed Description

Concrete cap class.

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

- C:/quantlib/QuantLib/ql/instruments/**capfloor.hpp**

7.43 QuantLib::CapFloor Class Reference

Base class for cap-like instruments.

`#include <capfloor.hpp>`

Inheritance diagram for QuantLib::CapFloor:

Collaboration diagram for QuantLib::CapFloor:

Classes

- class **arguments**
  Arguments for cap/floor calculation
- class **engine**
  base class for cap/floor engines

Public Types

- enum **Type** { **Cap**, **Floor**, **Collar** }

Public Member Functions

- **CapFloor** (Type type, const Leg &floatingLeg, const std::vector< Rate > &capRates, const std::vector< Rate > &floorRates)
- **CapFloor** (Type type, const Leg &floatingLeg, const std::vector< Rate > &strikes)
- Rate **atmRate** (const YieldTermStructure &discountCurve) const
- Volatility **impliedVolatility** (Real price, const Handle< YieldTermStructure > &disc, Volatility guess, Real accuracy=1.0e-4, Natural maxEvaluations=100, Volatility minVol=1.0e-7, Volatility maxVol=4.0, Real displacement=0.0) const

  implied term volatility

Instrument interface

- bool **isExpired** () const
- void **setupArguments** (PricingEngine::arguments *) const

Inspectors

- Type **type** () const
- const std::vector< Rate > & **capRates** () const
- const std::vector< Rate > & **floorRates** () const
- const Leg & **floatingLeg** () const
- Date **startDate** () const
- Date **maturityDate** () const
- boost::shared_ptr< FloatingRateCoupon > **lastFloatingRateCoupon** () const
- boost::shared_ptr< **CapFloor** > **optionlet** (const Size n) const

  *Returns the n-th optionlet as a new **CapFloor** (p. 53) with only one cash flow.*

7.43.1 Detailed Description

Base class for cap-like instruments.

Test

- the correctness of the returned value is tested by checking that the price of a cap (resp. floor) decreases (resp. increases) with the strike rate.
- the relationship between the values of caps, floors and the resulting collars is checked.
- the put-call parity between the values of caps, floors and swaps is checked.
- the correctness of the returned implied volatility is tested by using it for reproducing the target value.
- the correctness of the returned value is tested by checking it against a known good value.

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

- C:/quantlib/QuantLib/ql/instruments/**capfloor.hpp**
- C:/quantlib/QuantLib/ql/instruments/capfloor.cpp

7.44 QuantLib::CashOrNothingPayoff Class Reference

Binary cash-or-nothing payoff.

`#include <payoffs.hpp>`

Inheritance diagram for QuantLib::CashOrNothingPayoff:

Collaboration diagram for QuantLib::CashOrNothingPayoff:

Public Member Functions

- **CashOrNothingPayoff** (Option::Type type, Real strike, Real cashPayoff)
- Real **cashPayoff** () const

Payoff interface

- std::string **name** () const
- std::string **description** () const
- Real **operator()** (Real price) const
- virtual void **accept** (AcyclicVisitor &)

Protected Attributes

- Real **cashPayoff_**

Additional Inherited Members

7.44.1 Detailed Description

Binary cash-or-nothing payoff.

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

- C:/quantlib/QuantLib/ql/instruments/**payoffs.hpp**
- C:/quantlib/QuantLib/ql/instruments/payoffs.cpp

7.45 QuantLib::CCTEU Class Reference

`#include <btp.hpp>`

Inheritance diagram for QuantLib::CCTEU:

Collaboration diagram for QuantLib::CCTEU:

Public Member Functions

- **CCTEU** (const Date &maturityDate, Spread spread, const Handle< YieldTermStructure > &fwd↩ Curve=Handle< YieldTermStructure >(), const Date &startDate=Date(), const Date &issueDate=Date())

Bond interface

- Real **accruedAmount** (Date d=Date()) const
  accrued amount at a given date

Additional Inherited Members

7.45.1 Detailed Description

Italian **CCTEU** (p. 55) (Certificato di credito del tesoro) Euribor6M indexed floating rate bond

7.45.2 Member Function Documentation

7.45.2.1 Real QuantLib::CCTEU::accruedAmount (Date *d* = `Date()`) const `[inline]`, `[virtual]`

accrued amount at a given date

The default bond settlement is used if no date is given.

Reimplemented from **QuantLib::Bond** (p. 47).

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

- C:/quantlib/QuantLib/ql/instruments/bonds/**btp.hpp**
- C:/quantlib/QuantLib/ql/instruments/bonds/btp.cpp

7.46 QuantLib::Claim Class Reference

Claim (p. 56) associated to a default event.

`#include <claim.hpp>`

Inheritance diagram for QuantLib::Claim:

Collaboration diagram for QuantLib::Claim:

Public Member Functions

- virtual Real **amount** (const Date &defaultDate, Real notional, Real recoveryRate) const =0
- void **update** ()

7.46.1 Detailed Description

Claim (p. 56) associated to a default event.

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

- C:/quantlib/QuantLib/ql/instruments/**claim.hpp**

7.47 QuantLib::CliquetOption Class Reference

cliquet (Ratchet) option

`#include <cliquetoption.hpp>`

Inheritance diagram for QuantLib::CliquetOption:

Collaboration diagram for QuantLib::CliquetOption:

Classes

- class **arguments**
  Arguments for cliquet option calculation
- class **engine**
  Cliquet engine base class.

Public Member Functions

- **CliquetOption** (const boost::shared_ptr< **PercentageStrikePayoff** > &, const boost::shared_ptr< EuropeanExercise > &maturity, const std::vector< Date > &resetDates)
- void **setupArguments** (PricingEngine::arguments *) const

Additional Inherited Members

7.47.1 Detailed Description

cliquet (Ratchet) option

A cliquet option, also known as Ratchet option, is a series of forward-starting (a.k.a. deferred strike) options where the strike for each forward start option is set equal to a fixed percentage of the spot price at the beginning of each period.

Todo
- add local/global caps/floors
- add accrued coupon and last fixing

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

- C:/quantlib/QuantLib/ql/instruments/**cliquetoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/cliquetoption.cpp

7.48 QuantLib::CmsRateBond Class Reference

CMS-rate bond.

`#include <cmsratebond.hpp>`

Inheritance diagram for QuantLib::CmsRateBond:

Collaboration diagram for QuantLib::CmsRateBond:

Public Member Functions

- **CmsRateBond** (Natural settlementDays, Real faceAmount, const Schedule &schedule, const boost↩ ::shared_ptr< SwapIndex > &index, const DayCounter &paymentDayCounter, BusinessDayConvention paymentConvention=Following, Natural fixingDays=Null< Natural >(), const std::vector< Real > &gearings=std::vector< Real >(1, 1.0), const std::vector< Spread > &spreads=std::vector< Spread >(1, 0.0), const std::vector< Rate > &caps=std::vector< Rate >(), const std::vector< Rate > &floors=std↩ ::vector< Rate >(), bool inArrears=false, Real **redemption**=100.0, const Date &issueDate=Date())

Additional Inherited Members

7.48.1 Detailed Description

CMS-rate bond.

Test calculations are tested by checking results against cached values.

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

- C:/quantlib/QuantLib/ql/instruments/bonds/**cmsratebond.hpp**
- C:/quantlib/QuantLib/ql/instruments/bonds/cmsratebond.cpp

7.49 QuantLib::Collar Class Reference

Concrete collar class.

`#include <capfloor.hpp>`

Inheritance diagram for QuantLib::Collar:

Collaboration diagram for QuantLib::Collar:

Public Member Functions

- **Collar** (const Leg &floatingLeg, const std::vector< Rate > &capRates, const std::vector< Rate > &floor↩ Rates)

Additional Inherited Members

7.49.1 Detailed Description

Concrete collar class.

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

- C:/quantlib/QuantLib/ql/instruments/**capfloor.hpp**

7.50 QuantLib::CompositeInstrument Class Reference

Composite instrument

`#include <compositeinstrument.hpp>`

Inheritance diagram for QuantLib::CompositeInstrument:

Collaboration diagram for QuantLib::CompositeInstrument:

Public Member Functions

- void **add** (const boost::shared_ptr< Instrument > &instrument, Real multiplier=1.0)
  adds an instrument to the composite
- void **subtract** (const boost::shared_ptr< Instrument > &instrument, Real multiplier=1.0)
  shorts an instrument from the composite

Instrument interface

- bool **isExpired** () const
- void **performCalculations** () const

7.50.1 Detailed Description

Composite instrument

This instrument is an aggregate of other instruments. Its NPV is the sum of the NPVs of its components, each possibly multiplied by a given factor.

Example: static replication of a down-and-out barrier option (p. **??**)

Warning

Methods that drive the calculation directly (such as recalculate(), freeze() and others) might not work correctly.

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

- C:/quantlib/QuantLib/ql/instruments/**compositeinstrument.hpp**
- C:/quantlib/QuantLib/ql/instruments/compositeinstrument.cpp

7.51 QuantLib::ContinuousAveragingAsianOption Class Reference

Continuous-averaging Asian option.

`#include <asianoption.hpp>`

Inheritance diagram for QuantLib::ContinuousAveragingAsianOption:

Collaboration diagram for QuantLib::ContinuousAveragingAsianOption:

Classes

- class **arguments**

  Extra arguments for single-asset continuous-average Asian option.
- class **engine**

  Continuous-averaging Asian engine base class.

Public Member Functions

- **ContinuousAveragingAsianOption** (Average::Type averageType, const boost::shared_ptr< **StrikedType**↩ **Payoff** > &payoff, const boost::shared_ptr< Exercise > &exercise)
- void **setupArguments** (PricingEngine::arguments *) const

Protected Attributes

- Average::Type **averageType_**

Additional Inherited Members

7.51.1 Detailed Description

Continuous-averaging Asian option.

Todo add running average

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

- C:/quantlib/QuantLib/ql/instruments/**asianoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/asianoption.cpp

7.52 QuantLib::ContinuousFixedLookbackOption Class Reference

Continuous-fixed lookback option.

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousFixedLookbackOption:

Collaboration diagram for QuantLib::ContinuousFixedLookbackOption:

Classes

- class **arguments**

  Arguments for continuous fixed lookback option calculation
- class **engine**

  Continuous fixed lookback engine base class

Public Member Functions

- **ContinuousFixedLookbackOption** (Real currentMinmax, const boost::shared_ptr< **StrikedTypePayoff** > &payoff, const boost::shared_ptr< Exercise > &exercise)
- void **setupArguments** (PricingEngine::arguments *) const

Protected Attributes

- Real **minmax_**

Additional Inherited Members

7.52.1 Detailed Description

Continuous-fixed lookback option.

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/lookbackoption.cpp

7.53 QuantLib::ContinuousFloatingLookbackOption Class Reference

Continuous-floating lookback option.

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousFloatingLookbackOption:

Collaboration diagram for QuantLib::ContinuousFloatingLookbackOption:

Classes

- class **arguments**

  Arguments for continuous floating lookback option calculation
- class **engine**

  Continuous floating lookback engine base class

Public Member Functions

- **ContinuousFloatingLookbackOption** (Real currentMinmax, const boost::shared_ptr< **TypePayoff** > &payoff, const boost::shared_ptr< Exercise > &exercise)
- void **setupArguments** (PricingEngine::arguments *) const

Protected Attributes

- Real **minmax_**

Additional Inherited Members

7.53.1 Detailed Description

Continuous-floating lookback option.

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/lookbackoption.cpp

7.54 QuantLib::ContinuousPartialFixedLookbackOption Class Reference

Continuous-partial-fixed lookback option.

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousPartialFixedLookbackOption:

Collaboration diagram for QuantLib::ContinuousPartialFixedLookbackOption:

Classes

- class **arguments**

  Arguments for continuous partial fixed lookback option calculation
- class **engine**

  Continuous partial fixed lookback engine base class

Public Member Functions

- **ContinuousPartialFixedLookbackOption** (Date lookbackPeriodStart, const boost::shared_ptr< **Striked**↩ **TypePayoff** > &payoff, const boost::shared_ptr< Exercise > &exercise)
- void **setupArguments** (PricingEngine::arguments *) const

Protected Attributes

- Date **lookbackPeriodStart_**

Additional Inherited Members

7.54.1 Detailed Description

Continuous-partial-fixed lookback option.

From `http://help.rmetrics.org/fExoticOptions/LookbackOptions.html`:

For a partial-time fixed strike lookback option, the lookback period starts at a predetermined date after the initialization date of the option. The partial-time fixed strike lookback call option payoff is given by the difference between the maximum observed price of the underlying asset during the lookback period and the fixed strike price. The partial-time fixed strike lookback put option payoff is given by the difference between the fixed strike price and the minimum observed price of the underlying asset during the lookback period. The partial-time fixed strike lookback option is cheaper than a similar standard fixed strike lookback option. Partial-time fixed strike lookback options can be priced analytically using a model introduced by Heynen and Kat (1994).

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/lookbackoption.cpp

7.55 QuantLib::ContinuousPartialFloatingLookbackOption Class Reference

Continuous-partial-floating lookback option.

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousPartialFloatingLookbackOption:

Collaboration diagram for QuantLib::ContinuousPartialFloatingLookbackOption:

Classes

- class **arguments**

  Arguments for continuous partial floating lookback option calculation
- class **engine**

  Continuous partial floating lookback engine base class

Public Member Functions

- **ContinuousPartialFloatingLookbackOption** (Real currentMinmax, Real lambda, Date lookbackPeriodEnd, const boost::shared_ptr< **TypePayoff** > &payoff, const boost::shared_ptr< Exercise > &exercise)
- void **setupArguments** (PricingEngine::arguments *) const

Protected Attributes

- Real **lambda_**
- Date **lookbackPeriodEnd_**

Additional Inherited Members

7.55.1 Detailed Description

Continuous-partial-floating lookback option.

From `http://help.rmetrics.org/fExoticOptions/LookbackOptions.html` :

For a partial-time floating strike lookback option, the lookback period starts at time zero and ends at an arbitrary date before expiration. Except for the partial lookback period, the option is similar to a floating strike lookback option. The partial-time floating strike lookback option is cheaper than a similar standard floating strike lookback option. Partial-time floating strike lookback options can be priced analytically using a model introduced by Heynen and Kat (1994).

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/lookbackoption.cpp

7.56 QuantLib::CPIBond Class Reference

`#include <cpibond.hpp>`

Inheritance diagram for QuantLib::CPIBond:

Collaboration diagram for QuantLib::CPIBond:

Public Member Functions

- **CPIBond** (Natural settlementDays, Real faceAmount, bool growthOnly, Real baseCPI, const Period &observationLag, const boost::shared_ptr< ZeroInflationIndex > &cpiIndex, CPI::InterpolationType observationInterpolation, const Schedule &schedule, const std::vector< Rate > &coupons, const Day↩ Counter &accrualDayCounter, BusinessDayConvention paymentConvention=ModifiedFollowing, const Date &issueDate=Date(), const Calendar &paymentCalendar=Calendar(), const Period &exCoupon↩ Period=Period(), const Calendar &exCouponCalendar=Calendar(), const BusinessDayConvention ex↩ CouponConvention=Unadjusted, bool exCouponEndOfMonth=false)
- Frequency **frequency** () const
- const DayCounter & **dayCounter** () const
- bool **growthOnly** () const
- Real **baseCPI** () const
- Period **observationLag** () const
- const boost::shared_ptr< ZeroInflationIndex > & **cpiIndex** () const
- CPI::InterpolationType **observationInterpolation** () const

Protected Attributes

- Frequency **frequency_**
- DayCounter **dayCounter_**
- bool **growthOnly_**
- Real **baseCPI_**
- Period **observationLag_**
- boost::shared_ptr< ZeroInflationIndex > **cpiIndex_**
- CPI::InterpolationType **observationInterpolation_**

Additional Inherited Members

7.56.1 Detailed Description

cpi bond; if there is only one date in the schedule it is a zero bond returning an inflated notional.

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

- C:/quantlib/QuantLib/ql/instruments/bonds/**cpibond.hpp**
- C:/quantlib/QuantLib/ql/instruments/bonds/cpibond.cpp

7.57 QuantLib::CPICapFloor Class Reference

CPI cap or floor.

`#include <cpicapfloor.hpp>`

Inheritance diagram for QuantLib::CPICapFloor:

Collaboration diagram for QuantLib::CPICapFloor:

Classes

- class **arguments**
- class **engine**
- class **results**

Public Member Functions

- **CPICapFloor** (Option::Type type, Real nominal, const Date &startDate, Real baseCPI, const Date &maturity, const Calendar &fixCalendar, BusinessDayConvention fixConvention, const Calendar &payCalendar, BusinessDayConvention payConvention, Rate **strike**, const Handle< ZeroInflationIndex > &infIndex, const Period &observationLag, CPI::InterpolationType observationInterpolation=CPI::AsIndex)

Inspectors

- Option::Type **type** () const
- Real **nominal** () const
- Rate **strike** () const

  K *in the above formula.*
- Date **fixingDate** () const

  when you fix - but remember that there is an observation interpolation factor as well
- Date **payDate** () const
- Handle< ZeroInflationIndex > **inflationIndex** () const
- Period **observationLag** () const

Instrument interface

- bool **isExpired** () const
- void **setupArguments** (PricingEngine::arguments *) const
- void **fetchResults** (const PricingEngine::results *r) const

Protected Attributes

- Option::Type **type_**
- Real **nominal_**
- Date **startDate_**
- Date **fixDate_**
- Date **payDate_**
- Real **baseCPI_**
- Date **maturity_**
- Calendar **fixCalendar_**
- BusinessDayConvention **fixConvention_**
- Calendar **payCalendar_**
- BusinessDayConvention **payConvention_**
- Rate **strike_**
- Handle< ZeroInflationIndex > **inflndex_**
- Period **observationLag_**
- CPI::InterpolationType **observationInterpolation_**

7.57.1 Detailed Description

CPI cap or floor.

Quoted as a fixed strike rate K. Payoff:

$$P_n(0,T)\max(y(N[(1+K)^T-1]-N\left[\frac{I(T)}{I(0)}-1\right]),0)$$

where T is the maturity time, $P_n(0,t)$ is the nominal discount factor at time t, N is the notional, and $I(t)$ is the inflation index value at time t.

Inflation is generally available on every day, including holidays and weekends. Hence there is a variable to state whether the observe/fix dates for inflation are adjusted or not. The default is not to adjust.

N.B. a cpi cap or floor is an option, not a cap or floor on a coupon. Thus this is very similar to a ZCIIS and has a single flow, this is as usual for cpi because it is cumulative up to option maturity from base date.

We do not inherit from Option, although this would be reasonable, because we do not have that degree of generality.

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

- C:/quantlib/QuantLib/ql/instruments/**cpicapfloor.hpp**
- C:/quantlib/QuantLib/ql/instruments/cpicapfloor.cpp

7.58 QuantLib::CPISwap Class Reference

zero-inflation-indexed swap,

`#include <cpiswap.hpp>`

Inheritance diagram for QuantLib::CPISwap:

Collaboration diagram for QuantLib::CPISwap:

Classes

- class **arguments**

  Arguments for swap calculation
- class **engine**
- class **results**

  Results from swap calculation

Public Types

- enum **Type** { **Receiver** = -1, **Payer** = 1 }

Public Member Functions

- **CPISwap** (Type type, Real nominal, bool subtractInflationNominal, Spread spread, const DayCounter &floatDayCount, const Schedule &floatSchedule, const BusinessDayConvention &floatRoll, Natural fixing↩ Days, const boost::shared_ptr< IborIndex > &floatIndex, Rate fixedRate, Real baseCPI, const Day↩ Counter &fixedDayCount, const Schedule &fixedSchedule, const BusinessDayConvention &fixedRoll, const Period &observationLag, const boost::shared_ptr< ZeroInflationIndex > &fixedIndex, CPI::InterpolationType observationInterpolation=CPI::AsIndex, Real inflationNominal=Null< Real >())
- virtual Real **floatLegNPV** () const
- virtual Spread **fairSpread** () const
- virtual Real **fixedLegNPV** () const
- virtual Rate **fairRate** () const
- virtual Type **type** () const
- virtual Real **nominal** () const
- virtual bool **subtractInflationNominal** () const
- virtual Spread **spread** () const
- virtual const DayCounter & **floatDayCount** () const
- virtual const Schedule & **floatSchedule** () const
- virtual const BusinessDayConvention & **floatPaymentRoll** () const
- virtual Natural **fixingDays** () const
- virtual const boost::shared_ptr< IborIndex > & **floatIndex** () const
- virtual Rate **fixedRate** () const
- virtual Real **baseCPI** () const
- virtual const DayCounter & **fixedDayCount** () const
- virtual const Schedule & **fixedSchedule** () const
- virtual const BusinessDayConvention & **fixedPaymentRoll** () const
- virtual Period **observationLag** () const
- virtual const boost::shared_ptr< ZeroInflationIndex > & **fixedIndex** () const
- virtual CPI::InterpolationType **observationInterpolation** () const
- virtual Real **inflationNominal** () const
- virtual const Leg & **cpiLeg** () const
- virtual const Leg & **floatLeg** () const
- void **setupArguments** (PricingEngine::arguments *args) const

  for simple case sufficient to copy base class
- void **fetchResults** (const PricingEngine::results *) const

Additional Inherited Members

7.58.1 Detailed Description

zero-inflation-indexed swap,

fixed x zero-inflation, i.e. fixed x CPI(i'th fixing)/CPI(base) versus floating + spread

Note that this does ony the inflation-vs-floating-leg. Extension to inflation-vs-fixed-leg. is simple - just replace the floating leg with a fixed leg.

Typically there are notional exchanges at the end: either inflated-notional vs notional; or just (inflated-notional - notional) vs zero. The latter is perhaphs more typical.

Warning

Setting subtractInflationNominal to true means that the original inflation nominal is subtracted from both nominals before they are exchanged, even if they are different.

This swap can mimic a ZCIIS where [(1+q)^n - 1] is exchanged against (cpi ratio - 1), by using differnt nominals on each leg and setting subtractInflationNominal to true. ALSO - there must be just one date in each schedule.

The two legs can have different schedules, fixing (days vs lag), settlement, and roll conventions. N.B. accrual adjustment periods are already in the schedules. Trade date and swap settlement date are outside the scope of the instrument.

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

- C:/quantlib/QuantLib/ql/instruments/**cpiswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/cpiswap.cpp

7.59 QuantLib::CreditDefaultSwap Class Reference

Credit default swap.

`#include <creditdefaultswap.hpp>`

Inheritance diagram for QuantLib::CreditDefaultSwap:

Collaboration diagram for QuantLib::CreditDefaultSwap:

Classes

- class **arguments**
- class **engine**
- class **results**

Public Member Functions

Constructors

- **CreditDefaultSwap** (Protection::Side side, Real notional, Rate spread, const Schedule &schedule, BusinessDayConvention paymentConvention, const DayCounter &dayCounter, bool settlesAccrual=true, bool paysAtDefaultTime=true, const Date &protectionStart=Date(), const boost::shared_ptr< **Claim** > &=boost::shared_ptr< **Claim** >())

  CDS quoted as running-spread only.
- **CreditDefaultSwap** (Protection::Side side, Real notional, Rate upfront, Rate spread, const Schedule &schedule, BusinessDayConvention paymentConvention, const DayCounter &dayCounter, bool settles↩ Accrual=true, bool paysAtDefaultTime=true, const Date &protectionStart=Date(), const Date &upfront↩ Date=Date(), const boost::shared_ptr< **Claim** > &=boost::shared_ptr< **Claim** >())

  CDS quoted as upfront and running spread.

Inspectors

- Protection::Side **side** () const
- Real **notional** () const
- Rate **runningSpread** () const
- boost::optional< Rate > **upfront** () const
- bool **settlesAccrual** () const
- bool **paysAtDefaultTime** () const
- const Leg & **coupons** () const
- const Date & **protectionStartDate** () const

  The first date for which defaults will trigger the contract.
- const Date & **protectionEndDate** () const

  The last date for which defaults will trigger the contract.

Results

- Rate **fairUpfront** () const
- Rate **fairSpread** () const
- Real **couponLegBPS** () const
- Real **upfrontBPS** () const
- Real **couponLegNPV** () const
- Real **defaultLegNPV** () const
- Real **upfrontNPV** () const
- Rate **impliedHazardRate** (Real targetNPV, const Handle< YieldTermStructure > &discountCurve, const DayCounter &dayCounter, Real recoveryRate=0.4, Real accuracy=1.0e-6) const

  Implied hazard rate calculation.
- Rate **conventionalSpread** (Real conventionalRecovery, const Handle< YieldTermStructure > &discountCurve, const DayCounter &dayCounter) const

  Conventional/standard upfront-to-spread conversion.

Protected Attributes

- Protection::Side **side_**
- Real **notional_**
- boost::optional< Rate > **upfront_**
- Rate **runningSpread_**
- bool **settlesAccrual_**
- bool **paysAtDefaultTime_**
- boost::shared_ptr< **Claim** > **claim_**
- Leg **leg_**
- boost::shared_ptr< CashFlow > **upfrontPayment_**
- Date **protectionStart_**

- Rate **fairUpfront_**
- Rate **fairSpread_**
- Real **couponLegBPS_**
- Real **couponLegNPV_**
- Real **upfrontBPS_**
- Real **upfrontNPV_**
- Real **defaultLegNPV_**

Instrument interface

- bool **isExpired** () const
- void **setupArguments** (PricingEngine::arguments *) const
- void **fetchResults** (const PricingEngine::results *) const
- void **setupExpired** () const

7.59.1 Detailed Description

Credit default swap.

Note

This instrument currently assumes that the issuer did not default until today's date.

Warning

if `Settings::includeReferenceDateCashFlows()` is set to `true`, payments occurring at the settlement date of the swap might be included in the NPV and therefore affect the fair-spread calculation. This might not be what you want.

7.59.2 Constructor & Destructor Documentation

7.59.2.1 QuantLib::CreditDefaultSwap::CreditDefaultSwap (Protection::Side *side*, Real *notional*, Rate *spread*, const Schedule & *schedule*, BusinessDayConvention *paymentConvention*, const DayCounter & *dayCounter*, bool *settlesAccrual* = `true`, bool *paysAtDefaultTime* = `true`, const Date & *protectionStart* = `Date()`, const boost::shared_ptr< Claim > & *claim* = `boost::shared_ptr<Claim>()`)

CDS quoted as running-spread only.

Parameters

side	Whether the protection is bought or sold.
notional	Notional value
spread	Running spread in fractional units.
schedule	Coupon schedule.
paymentConvention	Business-day convention for payment-date adjustment.
dayCounter	Day-count convention for accrual.
settlesAccrual	Whether or not the accrued coupon is due in the event of a default.
paysAtDefaultTime	If set to true, any payments triggered by a default event are due at default time. If set to false, they are due at the end of the accrual period.
protectionStart	The first date where a default event will trigger the contract.

7.59.2.2 QuantLib::CreditDefaultSwap::CreditDefaultSwap (Protection::Side *side,* Real *notional,* Rate *upfront,* Rate *spread,* const Schedule & *schedule,* BusinessDayConvention *paymentConvention,* const DayCounter & *dayCounter,* bool *settlesAccrual* = `true`, bool *paysAtDefaultTime* = `true`, const Date & *protectionStart* = `Date()`, const Date & *upfrontDate* = `Date()`, const boost::shared_ptr< **Claim** > & *claim* = `boost::shared_ptr<`**Claim**`>()`)

CDS quoted as upfront and running spread.

Parameters

side	Whether the protection is bought or sold.
notional	Notional value
upfront	Upfront in fractional units.
spread	Running spread in fractional units.
schedule	Coupon schedule.
paymentConvention	Business-day convention for payment-date adjustment.
dayCounter	Day-count convention for accrual.
settlesAccrual	Whether or not the accrued coupon is due in the event of a default.
paysAtDefaultTime	If set to true, any payments triggered by a default event are due at default time. If set to false, they are due at the end of the accrual period.
protectionStart	The first date where a default event will trigger the contract.
upfrontDate	**Settlement** (p. 131) date for the upfront payment.

7.59.3 Member Function Documentation

7.59.3.1 Rate QuantLib::CreditDefaultSwap::conventionalSpread (Real *conventionalRecovery,* const Handle< YieldTermStructure > & *discountCurve,* const DayCounter & *dayCounter*) const

Conventional/standard upfront-to-spread conversion.

Under a standard ISDA model and a set of standardised instrument characteristics, it is the running only quoted spread that will make a CDS contract have an NPV of 0 when quoted for that running only spread. Refer to: "ISDA Standard CDS converter specification." May 2009.

The conventional recovery rate to apply in the calculation is as specified by ISDA, not necessarily equal to the market-quoted one. It is typically 0.4 for SeniorSec and 0.2 for subordinate.

Note

The conversion employs a flat hazard rate. As a result, you will not recover the market quotes.
This method performs the calculation with the instrument characteristics. It will coincide with the ISDA calculation if your object has the standard characteristics. Notably:

- The calendar should have no bank holidays, just weekends.
- The yield curve should be LIBOR piecewise constant in fwd rates, with a discount factor of 1 on the calculation date, which coincides with the trade date.
- Convention should be Following for yield curve and contract cashflows.
- The CDS should pay accrued and mature on standard IMM dates, settle on trade date +1 and upfront settle on trade date +3.

7.59.3.2 Real QuantLib::CreditDefaultSwap::couponLegBPS ( ) const

Returns the variation of the fixed-leg value given a one-basis-point change in the running spread.

7.59.3.3 Rate QuantLib::CreditDefaultSwap::fairSpread ( ) const

Returns the running spread that, given the quoted recovery rate, will make the running-only CDS have an NPV of 0.

Note

This calculation does not take any upfront into account, even if one was given.

7.59.3.4 Rate QuantLib::CreditDefaultSwap::fairUpfront ( ) const

Returns the upfront spread that, given the running spread and the quoted recovery rate, will make the instrument have an NPV of 0.

7.59.3.5 Rate QuantLib::CreditDefaultSwap::impliedHazardRate (Real *targetNPV,* const Handle< YieldTermStructure > & *discountCurve,* const DayCounter & *dayCounter,* Real *recoveryRate* = `0.4`, Real *accuracy* = `1.0e-6`) const

Implied hazard rate calculation.

Note

This method performs the calculation with the instrument characteristics. It will coincide with the ISDA calculation if your object has the standard characteristics. Notably:

- The calendar should have no bank holidays, just weekends.
- The yield curve should be LIBOR piecewise constant in fwd rates, with a discount factor of 1 on the calculation date, which coincides with the trade date.
- Convention should be Following for yield curve and contract cashflows.
- The CDS should pay accrued and mature on standard IMM dates, settle on trade date +1 and upfront settle on trade date +3.

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

- C:/quantlib/QuantLib/ql/instruments/**creditdefaultswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/creditdefaultswap.cpp

7.60 QuantLib::DiscreteAveragingAsianOption Class Reference

Discrete-averaging Asian option.

`#include <asianoption.hpp>`

Inheritance diagram for QuantLib::DiscreteAveragingAsianOption:

Collaboration diagram for QuantLib::DiscreteAveragingAsianOption:

Classes

- class **arguments**

  Extra arguments for single-asset discrete-average Asian option.
- class **engine**

  Discrete-averaging Asian engine base class.

Public Member Functions

- **DiscreteAveragingAsianOption** (Average::Type averageType, Real runningAccumulator, Size pastFixings, const std::vector< Date > &fixingDates, const boost::shared_ptr< **StrikedTypePayoff** > &payoff, const boost::shared_ptr< Exercise > &exercise)
- void **setupArguments** (PricingEngine::arguments *) const

Protected Attributes

- Average::Type **averageType_**
- Real **runningAccumulator_**
- Size **pastFixings_**
- std::vector< Date > **fixingDates_**

Additional Inherited Members

7.60.1 Detailed Description

Discrete-averaging Asian option.

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

- C:/quantlib/QuantLib/ql/instruments/**asianoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/asianoption.cpp

7.61 QuantLib::DividendBarrierOption Class Reference

Single-asset barrier option with discrete dividends.

`#include <dividendbarrieroption.hpp>`

Inheritance diagram for QuantLib::DividendBarrierOption:

Collaboration diagram for QuantLib::DividendBarrierOption:

Classes

- class **arguments**

  Arguments for dividend barrier option calculation
- class **engine**

  Dividend-barrier-option engine base class

Public Member Functions

- **DividendBarrierOption** (Barrier::Type barrierType, Real barrier, Real rebate, const boost::shared_ptr< **StrikedTypePayoff** > &payoff, const boost::shared_ptr< Exercise > &exercise, const std::vector< Date > &dividendDates, const std::vector< Real > &dividends)

Protected Member Functions

- void **setupArguments** (PricingEngine::arguments *) const

Additional Inherited Members

7.61.1 Detailed Description

Single-asset barrier option with discrete dividends.

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

- C:/quantlib/QuantLib/ql/instruments/**dividendbarrieroption.hpp**
- C:/quantlib/QuantLib/ql/instruments/dividendbarrieroption.cpp

7.62 QuantLib::DividendVanillaOption Class Reference

Single-asset vanilla option (no barriers) with discrete dividends.

`#include <dividendvanillaoption.hpp>`

Inheritance diagram for QuantLib::DividendVanillaOption:

Collaboration diagram for QuantLib::DividendVanillaOption:

Classes

- class **arguments**

  Arguments for dividend vanilla option calculation
- class **engine**

  Dividend-vanilla-option engine base class

Public Member Functions

- **DividendVanillaOption** (const boost::shared_ptr< **StrikedTypePayoff** > &payoff, const boost::shared_ptr< Exercise > &exercise, const std::vector< Date > &dividendDates, const std::vector< Real > &dividends)
- Volatility **impliedVolatility** (Real price, const boost::shared_ptr< GeneralizedBlackScholesProcess > &process, Real accuracy=1.0e-4, Size maxEvaluations=100, Volatility minVol=1.0e-7, Volatility maxVol=4.0) const

Protected Member Functions

- void **setupArguments** (PricingEngine::arguments *) const

Additional Inherited Members

7.62.1 Detailed Description

Single-asset vanilla option (no barriers) with discrete dividends.

7.62.2 Member Function Documentation

7.62.2.1 Volatility QuantLib::DividendVanillaOption::impliedVolatility (Real *price*, const boost::shared_ptr< GeneralizedBlackScholesProcess > & *process*, Real *accuracy* = 1.0e-4, Size *maxEvaluations* = 100, Volatility *minVol* = 1.0e-7, Volatility *maxVol* = 4.0) const

Warning

see **VanillaOption** (p. 140) for notes on implied-volatility calculation.

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

- C:/quantlib/QuantLib/ql/instruments/**dividendvanillaoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/dividendvanillaoption.cpp

7.63 QuantLib::DoubleStickyRatchetPayoff Class Reference

Intermediate class for single/double sticky/ratchet payoffs.

`#include <stickyratchet.hpp>`

Inheritance diagram for QuantLib::DoubleStickyRatchetPayoff:

Collaboration diagram for QuantLib::DoubleStickyRatchetPayoff:

Public Member Functions

- **DoubleStickyRatchetPayoff** (Real type1, Real type2, Real gearing1, Real gearing2, Real gearing3, Real spread1, Real spread2, Real spread3, Real initialValue1, Real initialValue2, Real accrualFactor)

Payoff interface

- std::string **name** () const
- Real **operator()** (Real forward) const
- std::string **description** () const
- virtual void **accept** (AcyclicVisitor &)

Protected Attributes

- Real **type1_**
- Real **type2_**
- Real **gearing1_**
- Real **gearing2_**
- Real **gearing3_**
- Real **spread1_**
- Real **spread2_**
- Real **spread3_**
- Real **initialValue1_**
- Real **initialValue2_**
- Real **accrualFactor_**

7.63.1 Detailed Description

Intermediate class for single/double sticky/ratchet payoffs.

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

- C:/quantlib/QuantLib/ql/instruments/**stickyratchet.hpp**
- C:/quantlib/QuantLib/ql/instruments/stickyratchet.cpp

7.64 QuantLib::CreditDefaultSwap::engine Class Reference

Inheritance diagram for QuantLib::CreditDefaultSwap::engine:

Collaboration diagram for QuantLib::CreditDefaultSwap::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**creditdefaultswap.hpp**

7.65 QuantLib::DividendBarrierOption::engine Class Reference

Dividend-barrier-option engine base class

`#include <dividendbarrieroption.hpp>`

Inheritance diagram for QuantLib::DividendBarrierOption::engine:

Collaboration diagram for QuantLib::DividendBarrierOption::engine:

7.65.1 Detailed Description

Dividend-barrier-option engine base class

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

- C:/quantlib/QuantLib/ql/instruments/**dividendbarrieroption.hpp**

7.66 QuantLib::DividendVanillaOption::engine Class Reference

Dividend-vanilla-option engine base class

`#include <dividendvanillaoption.hpp>`

Inheritance diagram for QuantLib::DividendVanillaOption::engine:

Collaboration diagram for QuantLib::DividendVanillaOption::engine:

7.66.1 Detailed Description

Dividend-vanilla-option engine base class

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

- C:/quantlib/QuantLib/ql/instruments/**dividendvanillaoption.hpp**

7.67 QuantLib::Swap::engine Class Reference

Inheritance diagram for QuantLib::Swap::engine:

Collaboration diagram for QuantLib::Swap::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**swap.hpp**

7.68 QuantLib::FloatFloatSwap::engine Class Reference

Inheritance diagram for QuantLib::FloatFloatSwap::engine:

Collaboration diagram for QuantLib::FloatFloatSwap::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**floatfloatswap.hpp**

7.69 QuantLib::Swaption::engine Class Reference

base class for swaption engines

`#include <swaption.hpp>`

Inheritance diagram for QuantLib::Swaption::engine:

Collaboration diagram for QuantLib::Swaption::engine:

7.69.1 Detailed Description

base class for swaption engines

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

- C:/quantlib/QuantLib/ql/instruments/**swaption.hpp**

7.70 QuantLib::YoYInflationCapFloor::engine Class Reference

base class for cap/floor engines

`#include <inflationcapfloor.hpp>`

Inheritance diagram for QuantLib::YoYInflationCapFloor::engine:

Collaboration diagram for QuantLib::YoYInflationCapFloor::engine:

7.70.1 Detailed Description

base class for cap/floor engines

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

- C:/quantlib/QuantLib/ql/instruments/inflationcapfloor.hpp

7.71 QuantLib::FloatFloatSwaption::engine Class Reference

base class for cms swaption engines

`#include <floatfloatswaption.hpp>`

Inheritance diagram for QuantLib::FloatFloatSwaption::engine:

Collaboration diagram for QuantLib::FloatFloatSwaption::engine:

7.71.1 Detailed Description

base class for cms swaption engines

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

- C:/quantlib/QuantLib/ql/instruments/**floatfloatswaption.hpp**

7.72 QuantLib::DiscreteAveragingAsianOption::engine Class Reference

Discrete-averaging Asian engine base class.

`#include <asianoption.hpp>`

Inheritance diagram for QuantLib::DiscreteAveragingAsianOption::engine:

Collaboration diagram for QuantLib::DiscreteAveragingAsianOption::engine:

7.72.1 Detailed Description

Discrete-averaging Asian engine base class.

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

- C:/quantlib/QuantLib/ql/instruments/**asianoption.hpp**

7.73 QuantLib::VanillaSwap::engine Class Reference

Inheritance diagram for QuantLib::VanillaSwap::engine:

Collaboration diagram for QuantLib::VanillaSwap::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**vanillaswap.hpp**

7.74 QuantLib::BasketOption::engine Class Reference

Basket-option engine base class

`#include <basketoption.hpp>`

Inheritance diagram for QuantLib::BasketOption::engine:

Collaboration diagram for QuantLib::BasketOption::engine:

7.74.1 Detailed Description

Basket-option engine base class

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

- C:/quantlib/QuantLib/ql/instruments/**basketoption.hpp**

7.75 QuantLib::VarianceSwap::engine Class Reference

base class for variance-swap engines

`#include <varianceswap.hpp>`

Inheritance diagram for QuantLib::VarianceSwap::engine:

Collaboration diagram for QuantLib::VarianceSwap::engine:

7.75.1 Detailed Description

base class for variance-swap engines

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

- C:/quantlib/QuantLib/ql/instruments/**varianceswap.hpp**

7.76 QuantLib::YearOnYearInflationSwap::engine Class Reference

Inheritance diagram for QuantLib::YearOnYearInflationSwap::engine:

Collaboration diagram for QuantLib::YearOnYearInflationSwap::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**yearonyearinflationswap.hpp**

7.77 QuantLib::ZeroCouponInflationSwap::engine Class Reference

Inheritance diagram for QuantLib::ZeroCouponInflationSwap::engine:

Collaboration diagram for QuantLib::ZeroCouponInflationSwap::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**zerocouponinflationswap.hpp**

7.78 QuantLib::CapFloor::engine Class Reference

base class for cap/floor engines

`#include <capfloor.hpp>`

Inheritance diagram for QuantLib::CapFloor::engine:

Collaboration diagram for QuantLib::CapFloor::engine:

7.78.1 Detailed Description

base class for cap/floor engines

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

- C:/quantlib/QuantLib/ql/instruments/**capfloor.hpp**

7.79 QuantLib::ContinuousAveragingAsianOption::engine Class Reference

Continuous-averaging Asian engine base class.

`#include <asianoption.hpp>`

Inheritance diagram for QuantLib::ContinuousAveragingAsianOption::engine:

Collaboration diagram for QuantLib::ContinuousAveragingAsianOption::engine:

7.79.1 Detailed Description

Continuous-averaging Asian engine base class.

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

- C:/quantlib/QuantLib/ql/instruments/**asianoption.hpp**

7.80 QuantLib::CliquetOption::engine Class Reference

Cliquet engine base class.

`#include <cliquetoption.hpp>`

Inheritance diagram for QuantLib::CliquetOption::engine:

Collaboration diagram for QuantLib::CliquetOption::engine:

7.80.1 Detailed Description

Cliquet engine base class.

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

- C:/quantlib/QuantLib/ql/instruments/**cliquetoption.hpp**

7.81 QuantLib::ContinuousFloatingLookbackOption::engine Class Reference

Continuous floating lookback engine base class

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousFloatingLookbackOption::engine:

Collaboration diagram for QuantLib::ContinuousFloatingLookbackOption::engine:

7.81.1 Detailed Description

Continuous floating lookback engine base class

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**

7.82 QuantLib::ContinuousFixedLookbackOption::engine Class Reference

Continuous fixed lookback engine base class

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousFixedLookbackOption::engine:

Collaboration diagram for QuantLib::ContinuousFixedLookbackOption::engine:

7.82.1 Detailed Description

Continuous fixed lookback engine base class

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**

7.83 QuantLib::Bond::engine Class Reference

Inheritance diagram for QuantLib::Bond::engine:

Collaboration diagram for QuantLib::Bond::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**bond.hpp**

7.84 QuantLib::ContinuousPartialFloatingLookbackOption::engine Class Reference

Continuous partial floating lookback engine base class

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousPartialFloatingLookbackOption::engine:

Collaboration diagram for QuantLib::ContinuousPartialFloatingLookbackOption::engine:

7.84.1 Detailed Description

Continuous partial floating lookback engine base class

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**

7.85 QuantLib::ContinuousPartialFixedLookbackOption::engine Class Reference

Continuous partial fixed lookback engine base class

`#include <lookbackoption.hpp>`

Inheritance diagram for QuantLib::ContinuousPartialFixedLookbackOption::engine:

Collaboration diagram for QuantLib::ContinuousPartialFixedLookbackOption::engine:

7.85.1 Detailed Description

Continuous partial fixed lookback engine base class

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

- C:/quantlib/QuantLib/ql/instruments/**lookbackoption.hpp**

7.86 QuantLib::BarrierOption::engine Class Reference

Barrier-option engine base class

`#include <barrieroption.hpp>`

Inheritance diagram for QuantLib::BarrierOption::engine:

Collaboration diagram for QuantLib::BarrierOption::engine:

Protected Member Functions

- bool **triggered** (Real underlying) const

7.86.1 Detailed Description

Barrier-option engine base class

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

- C:/quantlib/QuantLib/ql/instruments/**barrieroption.hpp**
- C:/quantlib/QuantLib/ql/instruments/barrieroption.cpp

7.87 QuantLib::CPICapFloor::engine Class Reference

Inheritance diagram for QuantLib::CPICapFloor::engine:

Collaboration diagram for QuantLib::CPICapFloor::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**cpicapfloor.hpp**

7.88 QuantLib::MultiAssetOption::engine Class Reference

Inheritance diagram for QuantLib::MultiAssetOption::engine:

Collaboration diagram for QuantLib::MultiAssetOption::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**multiassetoption.hpp**

7.89 QuantLib::NonstandardSwap::engine Class Reference

Inheritance diagram for QuantLib::NonstandardSwap::engine:

Collaboration diagram for QuantLib::NonstandardSwap::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**nonstandardswap.hpp**

7.90 QuantLib::CPISwap::engine Class Reference

Inheritance diagram for QuantLib::CPISwap::engine:

Collaboration diagram for QuantLib::CPISwap::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**cpiswap.hpp**

7.91 QuantLib::NonstandardSwaption::engine Class Reference

base class for nonstandard swaption engines

`#include <nonstandardswaption.hpp>`

Inheritance diagram for QuantLib::NonstandardSwaption::engine:

Collaboration diagram for QuantLib::NonstandardSwaption::engine:

7.91.1 Detailed Description

base class for nonstandard swaption engines

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

- C:/quantlib/QuantLib/ql/instruments/**nonstandardswaption.hpp**

7.92 QuantLib::OneAssetOption::engine Class Reference

Inheritance diagram for QuantLib::OneAssetOption::engine:

Collaboration diagram for QuantLib::OneAssetOption::engine:

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

- C:/quantlib/QuantLib/ql/instruments/**oneassetoption.hpp**

7.93 QuantLib::EuropeanOption Class Reference

European option on a single asset.

`#include <europeanoption.hpp>`

Inheritance diagram for QuantLib::EuropeanOption:

Collaboration diagram for QuantLib::EuropeanOption:

Public Member Functions

- **EuropeanOption** (const boost::shared_ptr< **StrikedTypePayoff** > &, const boost::shared_ptr< Exercise > &)

Additional Inherited Members

7.93.1 Detailed Description

European option on a single asset.

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

- C:/quantlib/QuantLib/ql/instruments/**europeanoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/europeanoption.cpp

7.94 QuantLib::FaceValueAccrualClaim Class Reference

Claim (p. 56) on the notional of a reference security, including accrual.

`#include <claim.hpp>`

Inheritance diagram for QuantLib::FaceValueAccrualClaim:

Collaboration diagram for QuantLib::FaceValueAccrualClaim:

Public Member Functions

- **FaceValueAccrualClaim** (const boost::shared_ptr< **Bond** > &referenceSecurity)
- Real **amount** (const Date &d, Real notional, Real recoveryRate) const

7.94.1 Detailed Description

Claim (p. 56) on the notional of a reference security, including accrual.

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

- C:/quantlib/QuantLib/ql/instruments/**claim.hpp**
- C:/quantlib/QuantLib/ql/instruments/claim.cpp

7.95 QuantLib::FaceValueClaim Class Reference

Claim (p. 56) on a notional.

`#include <claim.hpp>`

Inheritance diagram for QuantLib::FaceValueClaim:

Collaboration diagram for QuantLib::FaceValueClaim:

Public Member Functions

- Real **amount** (const Date &d, Real notional, Real recoveryRate) const

7.95.1 Detailed Description

Claim (p. 56) on a notional.

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

- C:/quantlib/QuantLib/ql/instruments/**claim.hpp**
- C:/quantlib/QuantLib/ql/instruments/claim.cpp

7.96 QuantLib::FixedRateBond Class Reference

fixed-rate bond

`#include <fixedratebond.hpp>`

Inheritance diagram for QuantLib::FixedRateBond:

Collaboration diagram for QuantLib::FixedRateBond:

Public Member Functions

- **FixedRateBond** (Natural settlementDays, Real faceAmount, const Schedule &schedule, const std↩ ::vector< Rate > &coupons, const DayCounter &accrualDayCounter, BusinessDayConvention payment↩ Convention=Following, Real **redemption**=100.0, const Date &issueDate=Date(), const Calendar &paymentCalendar=Calendar(), const Period &exCouponPeriod=Period(), const Calendar &exCoupon↩ Calendar=Calendar(), const BusinessDayConvention exCouponConvention=Unadjusted, bool exCoupon↩ EndOfMonth=false)

  simple annual compounding coupon rates
- **FixedRateBond** (Natural settlementDays, const Calendar &couponCalendar, Real faceAmount, const Date &startDate, const Date &maturityDate, const Period &tenor, const std::vector< Rate > &coupons, const DayCounter &accrualDayCounter, BusinessDayConvention accrualConvention=Following, BusinessDay↩ Convention paymentConvention=Following, Real **redemption**=100.0, const Date &issueDate=Date(), const Date &stubDate=Date(), DateGeneration::Rule rule=DateGeneration::Backward, bool endOfMonth=false, const Calendar &paymentCalendar=Calendar(), const Period &exCouponPeriod=Period(), const Calendar &exCouponCalendar=Calendar(), const BusinessDayConvention exCouponConvention=Unadjusted, bool exCouponEndOfMonth=false)
- **FixedRateBond** (Natural settlementDays, Real faceAmount, const Schedule &schedule, const std↩ ::vector< InterestRate > &coupons, BusinessDayConvention paymentConvention=Following, Real **redemption**=100.0, const Date &issueDate=Date(), const Calendar &paymentCalendar=Calendar(), const Period &exCouponPeriod=Period(), const Calendar &exCouponCalendar=Calendar(), const BusinessDay↩ Convention exCouponConvention=Unadjusted, bool exCouponEndOfMonth=false)

  generic compounding and frequency InterestRate coupons
- Frequency **frequency** () const
- const DayCounter & **dayCounter** () const

Protected Attributes

- Frequency **frequency_**
- DayCounter **dayCounter_**

Additional Inherited Members

7.96.1 Detailed Description

fixed-rate bond

Test calculations are tested by checking results against cached values.

7.96.2 Constructor & Destructor Documentation

7.96.2.1 **QuantLib::FixedRateBond::FixedRateBond (Natural *settlementDays,* const Calendar & *couponCalendar,* Real *faceAmount,* const Date & *startDate,* const Date & *maturityDate,* const Period & *tenor,* const std::vector< Rate > & *coupons,* const DayCounter & *accrualDayCounter,* BusinessDayConvention *accrualConvention* =** `Following`**,** BusinessDayConvention *paymentConvention* =** `Following`**,** Real *redemption* =** `100.0`**,** const Date & *issueDate* =** `Date()`**,** const Date & *stubDate* =** `Date()`**,** DateGeneration::Rule *rule* =** `DateGeneration::Backward`**,** bool *endOfMonth* =** `false`**,** const Calendar & *paymentCalendar* =** `Calendar()`**,** const Period & *exCouponPeriod* =** `Period()`**,** const Calendar & *exCouponCalendar* =** `Calendar()`**,** const BusinessDayConvention *exCouponConvention* =** `Unadjusted`**,** bool *exCouponEndOfMonth* =** `false` **)**

simple annual compounding coupon rates with internal schedule calculation

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

- C:/quantlib/QuantLib/ql/instruments/bonds/**fixedratebond.hpp**
- C:/quantlib/QuantLib/ql/instruments/bonds/fixedratebond.cpp

7.97 QuantLib::FixedRateBondForward Class Reference

Forward contract on a fixed-rate bond

`#include <fixedratebondforward.hpp>`

Inheritance diagram for QuantLib::FixedRateBondForward:

Collaboration diagram for QuantLib::FixedRateBondForward:

Public Member Functions

Constructors

- **FixedRateBondForward** (const Date &valueDate, const Date &maturityDate, Position::Type type, Real strike, Natural settlementDays, const DayCounter &dayCounter, const Calendar &calendar, BusinessDayConvention businessDayConvention, const boost::shared_ptr< **FixedRateBond** > &fixedCouponBond, const Handle< YieldTermStructure > &**discountCurve**=Handle< YieldTermStructure >(), const Handle< YieldTermStructure > &**incomeDiscountCurve**=Handle< YieldTermStructure >())

Calculations

- Real **forwardPrice** () const
  (dirty) forward bond price
- Real **cleanForwardPrice** () const
  (dirty) forward bond price minus accrued on bond at delivery
- Real **spotIncome** (const Handle< YieldTermStructure > &**incomeDiscountCurve**) const
  NPV of bond coupons discounted using incomeDiscountCurve.
- Real **spotValue** () const
  NPV of underlying bond.

Protected Member Functions

- void **performCalculations** () const

Protected Attributes

- boost::shared_ptr< **FixedRateBond** > **fixedCouponBond_**

7.97.1 Detailed Description

Forward contract on a fixed-rate bond

1. valueDate refers to the settlement date of the bond forward contract. maturityDate is the delivery (or repurchase) date for the underlying bond (not the bond's maturity date).
2. Relevant formulas used in the calculations (P refers to a price):
   a. $P_{CleanFwd}(t) = P_{DirtyFwd}(t) - AI(t = deliveryDate)$ where AI refers to the accrued interest on the underlying bond.
   b. $P_{DirtyFwd}(t) = \frac{P_{DirtySpot}(t) - SpotIncome(t)}{discountCurve->discount(t=deliveryDate)}$
   c. $SpotIncome(t) = \sum_i (CF_i \times incomeDiscountCurve-> discount(t_i))$ where CF_i represents the ith bond cash flow (coupon payment) associated with the underlying bond falling between the settlementDate and the deliveryDate. (Note the two different discount curves used in b. and c.)

Example: valuation of a repo on a fixed-rate bond (p. **??**)

Todo Add preconditions and tests

Todo Create switch- if coupon goes to seller is toggled on, don't consider income in the $P_{DirtyFwd}(t)$ calculation.

Todo Verify this works when the underlying is paper (in which case ignore all AI.)

Warning

This class still needs to be rigorously tested

7.97.2 Constructor & Destructor Documentation

7.97.2.1 QuantLib::FixedRateBondForward::FixedRateBondForward (const Date & *valueDate,* const Date & *maturityDate,* Position::Type *type,* Real *strike,* Natural *settlementDays,* const DayCounter & *dayCounter,* const Calendar & *calendar,* BusinessDayConvention *businessDayConvention,* const boost::shared_ptr< **FixedRateBond** > & *fixedCouponBond,* const Handle< YieldTermStructure > & *discountCurve =* `Handle<YieldTermStructure>()`, const Handle< YieldTermStructure > & *incomeDiscountCurve =* `Handle<YieldTermStructure>()`)

If strike is given in the constructor, can calculate the NPV of the contract via NPV().

If strike/forward price is desired, it can be obtained via **forwardPrice()** (p. 89). In this case, the strike variable in the constructor is irrelevant and will be ignored.

7.97.3 Member Function Documentation

7.97.3.1 Real QuantLib::FixedRateBondForward::spotIncome (const Handle< YieldTermStructure > & *incomeDiscountCurve*) const `[virtual]`

NPV of bond coupons discounted using incomeDiscountCurve.

Here only coupons between max(evaluation date,settlement date) and maturity date of bond forward contract are considered income.

Implements **QuantLib::Forward** (p. 95).

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

- C:/quantlib/QuantLib/ql/instruments/**fixedratebondforward.hpp**
- C:/quantlib/QuantLib/ql/instruments/fixedratebondforward.cpp

7.98 QuantLib::FloatFloatSwap Class Reference

float float swap

`#include <floatfloatswap.hpp>`

Inheritance diagram for QuantLib::FloatFloatSwap:

Collaboration diagram for QuantLib::FloatFloatSwap:

Classes

- class **arguments**

  Arguments for float float swap calculation
- class **engine**
- class **results**

  Results from float float swap calculation

Public Member Functions

- **FloatFloatSwap** (const VanillaSwap::Type type, const Real nominal1, const Real nominal2, const Schedule &schedule1, const boost::shared_ptr< InterestRateIndex > &index1, const DayCounter &dayCount1, const Schedule &schedule2, const boost::shared_ptr< InterestRateIndex > &index2, const DayCounter &dayCount2, const bool intermediateCapitalExchange=false, const bool finalCapitalExchange=false, const Real gearing1=1.0, const Real spread1=0.0, const Real cappedRate1=Null< Real >(), const Real floored↩ Rate1=Null< Real >(), const Real gearing2=1.0, const Real spread2=0.0, const Real cappedRate2=Null< Real >(), const Real flooredRate2=Null< Real >(), boost::optional< BusinessDayConvention > payment↩ Convention1=boost::none, boost::optional< BusinessDayConvention > paymentConvention2=boost::none)
- **FloatFloatSwap** (const VanillaSwap::Type type, const std::vector< Real > &nominal1, const std::vector< Real > &nominal2, const Schedule &schedule1, const boost::shared_ptr< InterestRateIndex > &index1, const DayCounter &dayCount1, const Schedule &schedule2, const boost::shared_ptr< InterestRate↩ Index > &index2, const DayCounter &dayCount2, const bool intermediateCapitalExchange=false, const bool finalCapitalExchange=false, const std::vector< Real > &gearing1=std::vector< Real >(), const std↩ ::vector< Real > &spread1=std::vector< Real >(), const std::vector< Real > &cappedRate1=std::vector< Real >(), const std::vector< Real > &flooredRate1=std::vector< Real >(), const std::vector< Real > &gearing2=std::vector< Real >(), const std::vector< Real > &spread2=std::vector< Real >(), const std↩ ::vector< Real > &cappedRate2=std::vector< Real >(), const std::vector< Real > &flooredRate2=std↩ ::vector< Real >(), boost::optional< BusinessDayConvention > paymentConvention1=boost::none, boost↩ ::optional< BusinessDayConvention > paymentConvention2=boost::none)
- void **setupArguments** (PricingEngine::arguments *args) const
- void **fetchResults** (const PricingEngine::results *) const

Inspectors

- VanillaSwap::Type **type** () const
- const std::vector< Real > & **nominal1** () const
- const std::vector< Real > & **nominal2** () const
- const Schedule & **schedule1** () const
- const Schedule & **schedule2** () const
- const boost::shared_ptr< InterestRateIndex > & **index1** () const
- const boost::shared_ptr< InterestRateIndex > & **index2** () const
- const std::vector< Real > **spread1** () const
- const std::vector< Real > **spread2** () const
- const std::vector< Real > **gearing1** () const
- const std::vector< Real > **gearing2** () const
- const std::vector< Rate > **cappedRate1** () const
- const std::vector< Rate > **flooredRate1** () const
- const std::vector< Rate > **cappedRate2** () const
- const std::vector< Rate > **flooredRate2** () const
- const DayCounter & **dayCount1** () const
- const DayCounter & **dayCount2** () const
- BusinessDayConvention **paymentConvention1** () const
- BusinessDayConvention **paymentConvention2** () const
- const Leg & **leg1** () const
- const Leg & **leg2** () const

Additional Inherited Members

7.98.1 Detailed Description

float float swap

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

- C:/quantlib/QuantLib/ql/instruments/**floatfloatswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/floatfloatswap.cpp

7.99 QuantLib::FloatFloatSwaption Class Reference

floatfloat swaption class

`#include <floatfloatswaption.hpp>`

Inheritance diagram for QuantLib::FloatFloatSwaption:

Collaboration diagram for QuantLib::FloatFloatSwaption:

Classes

- class **arguments**

  Arguments for cms swaption calculation
- class **engine**

  base class for cms swaption engines

Public Member Functions

- **FloatFloatSwaption** (const boost::shared_ptr< **FloatFloatSwap** > &swap, const boost::shared_ptr< Exercise > &exercise)
- Disposable< std::vector< boost::shared_ptr< CalibrationHelper > > > **calibrationBasket** (boost::shared_ptr< SwapIndex > standardSwapBase, boost::shared_ptr< SwaptionVolatilityStructure > swaptionVolatility, const BasketGeneratingEngine::CalibrationBasketType basketType=BasketGeneratingEngine::MaturityStrikeByDeltaGamma) const

Instrument interface

- bool **isExpired** () const
- void **setupArguments** (PricingEngine::arguments *) const

Inspectors

- VanillaSwap::Type **type** () const
- const boost::shared_ptr< **FloatFloatSwap** > & **underlyingSwap** () const

7.99.1 Detailed Description

floatfloat swaption class

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

- C:/quantlib/QuantLib/ql/instruments/**floatfloatswaption.hpp**
- C:/quantlib/QuantLib/ql/instruments/floatfloatswaption.cpp

7.100 QuantLib::FloatingRateBond Class Reference

floating-rate bond (possibly capped and/or floored)

`#include <floatingratebond.hpp>`

Inheritance diagram for QuantLib::FloatingRateBond:

Collaboration diagram for QuantLib::FloatingRateBond:

Public Member Functions

- **FloatingRateBond** (Natural settlementDays, Real faceAmount, const Schedule &schedule, const boost::shared_ptr< IborIndex > &iborIndex, const DayCounter &accrualDayCounter, BusinessDay↩ Convention paymentConvention=Following, Natural fixingDays=Null< Natural >(), const std::vector< Real > &gearings=std::vector< Real >(1, 1.0), const std::vector< Spread > &spreads=std::vector< Spread >(1, 0.0), const std::vector< Rate > &caps=std::vector< Rate >(), const std::vector< Rate > &floors=std↩ ::vector< Rate >(), bool inArrears=false, Real **redemption**=100.0, const Date &issueDate=Date())
- **FloatingRateBond** (Natural settlementDays, Real faceAmount, const Date &startDate, const Date &maturityDate, Frequency couponFrequency, const Calendar &calendar, const boost::shared_ptr< IborIndex > &iborIndex, const DayCounter &accrualDayCounter, BusinessDayConvention accrual↩ Convention=Following, BusinessDayConvention paymentConvention=Following, Natural fixingDays=Null< Natural >(), const std::vector< Real > &gearings=std::vector< Real >(1, 1.0), const std::vector< Spread > &spreads=std::vector< Spread >(1, 0.0), const std::vector< Rate > &caps=std::vector< Rate >(), const std::vector< Rate > &floors=std::vector< Rate >(), bool inArrears=false, Real **redemption**=100.0, const Date &issueDate=Date(), const Date &stubDate=Date(), DateGeneration::Rule rule=DateGeneration::↩ Backward, bool endOfMonth=false)

Additional Inherited Members

7.100.1 Detailed Description

floating-rate bond (possibly capped and/or floored)

Test calculations are tested by checking results against cached values.

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

- C:/quantlib/QuantLib/ql/instruments/bonds/**floatingratebond.hpp**
- C:/quantlib/QuantLib/ql/instruments/bonds/floatingratebond.cpp

7.101 QuantLib::FloatingTypePayoff Class Reference

Payoff based on a floating strike

`#include <payoffs.hpp>`

Inheritance diagram for QuantLib::FloatingTypePayoff:

Collaboration diagram for QuantLib::FloatingTypePayoff:

Public Member Functions

- **FloatingTypePayoff** (Option::Type type)

Payoff interface

- std::string **name** () const
- Real **operator()** (Real price) const
- virtual void **accept** (AcyclicVisitor &)

Additional Inherited Members

7.101.1 Detailed Description

Payoff based on a floating strike

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

- C:/quantlib/QuantLib/ql/instruments/**payoffs.hpp**
- C:/quantlib/QuantLib/ql/instruments/payoffs.cpp

7.102 QuantLib::Floor Class Reference

Concrete floor class.

`#include <capfloor.hpp>`

Inheritance diagram for QuantLib::Floor:

Collaboration diagram for QuantLib::Floor:

Public Member Functions

- **Floor** (const Leg &floatingLeg, const std::vector< Rate > &exerciseRates)

Additional Inherited Members

7.102.1 Detailed Description

Concrete floor class.

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

- C:/quantlib/QuantLib/ql/instruments/**capfloor.hpp**

7.103 QuantLib::Forward Class Reference

Abstract base forward class.

`#include <forward.hpp>`

Inheritance diagram for QuantLib::Forward:

Collaboration diagram for QuantLib::Forward:

Public Member Functions

- virtual Real **spotValue** () const =0
  returns spot value/price of an underlying financial instrument
- virtual Real **spotIncome** (const Handle< YieldTermStructure > &**incomeDiscountCurve**) const =0
  NPV of income/dividends/storage-costs etc. of underlying instrument.

Inspectors

- virtual Date **settlementDate** () const
- const Calendar & **calendar** () const
- BusinessDayConvention **businessDayConvention** () const
- const DayCounter & **dayCounter** () const
- Handle< YieldTermStructure > **discountCurve** () const
  term structure relevant to the contract (e.g. repo curve)
- Handle< YieldTermStructure > **incomeDiscountCurve** () const
  term structure that discounts the underlying's income cash flows
- bool **isExpired** () const
  returns whether the instrument is still tradable.

Calculations

- virtual Real **forwardValue** () const
  forward value/price of underlying, discounting income/dividends
- InterestRate **impliedYield** (Real underlyingSpotValue, Real **forwardValue**, Date settlementDate, Compounding compoundingConvention, DayCounter dayCounter)

Protected Member Functions

- **Forward** (const DayCounter &dayCounter, const Calendar &calendar, BusinessDayConvention business↩ DayConvention, Natural settlementDays, const boost::shared_ptr< Payoff > &payoff, const Date &valueDate, const Date &maturityDate, const Handle< YieldTermStructure > &**discountCurve**=Handle< YieldTerm↩ Structure >())
- void **performCalculations** () const

Protected Attributes

- Real **underlyingIncome_**
- Real **underlyingSpotValue_**
- DayCounter **dayCounter_**
- Calendar **calendar_**
- BusinessDayConvention **businessDayConvention_**
- Natural **settlementDays_**
- boost::shared_ptr< Payoff > **payoff_**
- Date **valueDate_**
- Date **maturityDate_**
  maturityDate of the forward contract or delivery date of underlying
- Handle< YieldTermStructure > **discountCurve_**
- Handle< YieldTermStructure > **incomeDiscountCurve_**

7.103.1 Detailed Description

Abstract base forward class.

Derived classes must implement the virtual functions **spotValue()** (p. 95) (NPV or spot price) and **spotIncome()** (p. 95) associated with the specific relevant underlying (e.g. bond, stock, commodity, loan/deposit). These functions must be used to set the protected member variables underlyingSpotValue_ and underlyingIncome_ within performCalculations() in the derived class before the base-class implementation is called.

spotIncome() (p. 95) refers generically to the present value of coupons, dividends or storage costs.

discountCurve_ is the curve used to discount forward contract cash flows back to the evaluation day, as well as to obtain forward values for spot values/prices.

incomeDiscountCurve_, which for generality is not automatically set to the discountCurve_, is the curve used to discount future income/dividends/storage-costs etc back to the evaluation date.

Todo Add preconditions and tests

Warning

This class still needs to be rigorously tested

7.103.2 Member Function Documentation

7.103.2.1 Real QuantLib::Forward::forwardValue ( ) const [virtual]

forward value/price of underlying, discounting income/dividends

Note

if this is a bond forward price, is must be a dirty forward price.

7.103.2.2 InterestRate QuantLib::Forward::impliedYield (Real *underlyingSpotValue,* Real *forwardValue,* Date *settlementDate,* Compounding *compoundingConvention,* DayCounter *dayCounter*)

Simple yield calculation based on underlying spot and forward values, taking into account underlying income. When $t > 0$, call with: underlyingSpotValue=spotValue(t), forwardValue=strikePrice, to get current yield. For a repo, if $t = 0$, impliedYield should reproduce the spot repo rate. For FRA's, this should reproduce the relevant zero rate at the FRA's maturityDate_;

7.103.3 Member Data Documentation

7.103.3.1 Handle<YieldTermStructure> QuantLib::Forward::incomeDiscountCurve_ [protected]

must set this in derived classes, based on particular underlying

7.103.3.2 Real QuantLib::Forward::underlyingIncome_ [mutable], [protected]

derived classes must set this, typically via **spotIncome()** (p. 95)

7.103.3.3 Real QuantLib::Forward::underlyingSpotValue_ [mutable], [protected]

derived classes must set this, typically via **spotValue()** (p. 95)

7.103.3.4 Date QuantLib::Forward::valueDate_ [protected]

valueDate = settlement date (date the fwd contract starts accruing)

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

- C:/quantlib/QuantLib/ql/instruments/**forward.hpp**
- C:/quantlib/QuantLib/ql/instruments/forward.cpp

7.104 QuantLib::ForwardOptionArguments< ArgumentsType > Class Template Reference

Arguments for forward (strike-resetting) option calculation

`#include <forwardvanillaoption.hpp>`

Inheritance diagram for QuantLib::ForwardOptionArguments< ArgumentsType >:

Collaboration diagram for QuantLib::ForwardOptionArguments< ArgumentsType >:

Public Member Functions

- void **validate** () const

Public Attributes

- Real **moneyness**
- Date **resetDate**

7.104.1 Detailed Description

template<class ArgumentsType>
class QuantLib::ForwardOptionArguments< ArgumentsType >

Arguments for forward (strike-resetting) option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**forwardvanillaoption.hpp**

7.105 QuantLib::ForwardRateAgreement Class Reference

Inheritance diagram for QuantLib::ForwardRateAgreement:

Collaboration diagram for QuantLib::ForwardRateAgreement:

Public Member Functions

- **ForwardRateAgreement** (const Date &valueDate, const Date &maturityDate, Position::Type type, Rate strikeForwardRate, Real notionalAmount, const boost::shared_ptr< IborIndex > &index, const Handle< YieldTermStructure > &**discountCurve**=Handle< YieldTermStructure >())

Calculations

- bool **isExpired** () const
- Date **settlementDate** () const
- Real **spotIncome** (const Handle< YieldTermStructure > &**incomeDiscountCurve**) const
- Real **spotValue** () const

  Spot value (NPV) of the underlying loan.
- InterestRate **forwardRate** () const

  Returns the relevant forward rate associated with the FRA term.

Protected Member Functions

- void **performCalculations** () const

Protected Attributes

- Position::Type **fraType_**
- InterestRate **forwardRate_**

  aka FRA rate (the market forward rate)
- InterestRate **strikeForwardRate_**

  aka FRA fixing rate, contract rate
- Real **notionalAmount_**
- boost::shared_ptr< IborIndex > **index_**

7.105.1 Member Function Documentation

7.105.1.1 bool QuantLib::ForwardRateAgreement::isExpired ( ) const

A FRA expires/settles on the valueDate

7.105.1.2 Date QuantLib::ForwardRateAgreement::settlementDate ( ) const [virtual]

This returns evaluationDate + settlementDays (not FRA valueDate).

Reimplemented from **QuantLib::Forward** (p. 95).

7.105.1.3 Real QuantLib::ForwardRateAgreement::spotIncome (const Handle< YieldTermStructure > & *incomeDiscountCurve*) const [virtual]

Income is zero for a FRA

Implements **QuantLib::Forward** (p. 95).

7.105.1.4 Real QuantLib::ForwardRateAgreement::spotValue ( ) const [virtual]

Spot value (NPV) of the underlying loan.

This has always a positive value (asset), even if short the FRA

Implements **QuantLib::Forward** (p. 95).

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

- C:/quantlib/QuantLib/ql/instruments/**forwardrateagreement.hpp**
- C:/quantlib/QuantLib/ql/instruments/forwardrateagreement.cpp

7.106 QuantLib::ForwardTypePayoff Class Reference

Class for forward type payoffs.

`#include <forward.hpp>`

Inheritance diagram for QuantLib::ForwardTypePayoff:

Collaboration diagram for QuantLib::ForwardTypePayoff:

Public Member Functions

- **ForwardTypePayoff** (Position::Type type, Real strike)
- Position::Type **forwardType** () const
- Real **strike** () const

Payoff interface

- std::string **name** () const
- std::string **description** () const
- Real **operator()** (Real price) const

Protected Attributes

- Position::Type **type_**
- Real **strike_**

7.106.1 Detailed Description

Class for forward type payoffs.

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

- C:/quantlib/QuantLib/ql/instruments/**forward.hpp**

7.107 QuantLib::ForwardVanillaOption Class Reference

Forward version of a vanilla option

`#include <forwardvanillaoption.hpp>`

Inheritance diagram for QuantLib::ForwardVanillaOption:

Collaboration diagram for QuantLib::ForwardVanillaOption:

Public Types

- typedef **ForwardOptionArguments**< OneAssetOption::arguments > **arguments**
- typedef **OneAssetOption::results results**

Public Member Functions

- **ForwardVanillaOption** (Real moneyness, const Date &resetDate, const boost::shared_ptr< **StrikedType**↩ **Payoff** > &payoff, const boost::shared_ptr< Exercise > &exercise)
- void **setupArguments** (PricingEngine::arguments *) const
- void **fetchResults** (const PricingEngine::results *) const

Additional Inherited Members

7.107.1 Detailed Description

Forward version of a vanilla option

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

- C:/quantlib/QuantLib/ql/instruments/**forwardvanillaoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/forwardvanillaoption.cpp

7.108 QuantLib::Futures Struct Reference

Public Types

- enum **Type** { **IMM**, **ASX** }

  Futures *(*p. *100) type enumeration.*

Related Functions

(Note that these are not member functions.)

- std::ostream & **operator**<< (std::ostream &, **Futures::Type**)

7.108.1 Member Enumeration Documentation

7.108.1.1 enum QuantLib::Futures::Type

Futures (p. 100) type enumeration.

These conventions specify the kind of futures type.

Enumerator

IMM Chicago Mercantile Internation Money Market, i.e. third Wednesday of March, June, September, December

ASX Australian Security Exchange, i.e. second Friday of March, June, September, December

7.108.2 Friends And Related Function Documentation

7.108.2.1 std::ostream & operator<< (std::ostream & , Futures::Type) [related]

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

- C:/quantlib/QuantLib/ql/instruments/**futures.hpp**

7.109 QuantLib::GapPayoff Class Reference

Binary gap payoff.

`#include <payoffs.hpp>`

Inheritance diagram for QuantLib::GapPayoff:

Collaboration diagram for QuantLib::GapPayoff:

Public Member Functions

- **GapPayoff** (Option::Type type, Real strike, Real secondStrike)
- Real **secondStrike** () const

Payoff interface

- std::string **name** () const
- std::string **description** () const
- Real **operator()** (Real price) const
- virtual void **accept** (AcyclicVisitor &)

Protected Attributes

- Real **secondStrike_**

Additional Inherited Members

7.109.1 Detailed Description

Binary gap payoff.

This payoff is equivalent to being a) long a **PlainVanillaPayoff** (p. 115) at the first strike (same Call/Put type) and b) short a **CashOrNothingPayoff** (p. 55) at the first strike (same Call/Put type) with cash payoff equal to the difference between the second and the first strike.

Warning

this payoff can be negative depending on the strikes

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

- C:/quantlib/QuantLib/ql/instruments/**payoffs.hpp**
- C:/quantlib/QuantLib/ql/instruments/payoffs.cpp

7.110 QuantLib::detail::ImpliedVolatilityHelper Class Reference

helper class for one-asset implied-volatility calculation

`#include <impliedvolatility.hpp>`

Static Public Member Functions

- static Volatility **calculate** (const Instrument &instrument, const PricingEngine &engine, SimpleQuote &vol↩ Quote, Real targetValue, Real accuracy, Natural maxEvaluations, Volatility minVol, Volatility maxVol)
- static boost::shared_ptr< GeneralizedBlackScholesProcess > **clone** (const boost::shared_ptr< Generalized↩ BlackScholesProcess > &, const boost::shared_ptr< SimpleQuote > &)

7.110.1 Detailed Description

helper class for one-asset implied-volatility calculation

The passed engine must be linked to the passed quote (see, e.g., **VanillaOption** (p. 140) to see how this can be achieved.)

Note

this function is meant for developers of option classes so that they can implement an impliedVolatility() method.

7.110.2 Member Function Documentation

7.110.2.1 boost::shared_ptr< GeneralizedBlackScholesProcess > QuantLib::detail::ImpliedVolatilityHelper::clone (const boost::shared_ptr< GeneralizedBlackScholesProcess > & *process,* const boost::shared_ptr< SimpleQuote > & *volQuote*) [static]

The returned process is equal to the passed one, except for the volatility which is flat and whose value is driven by the passed quote.

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

- C:/quantlib/QuantLib/ql/instruments/**impliedvolatility.hpp**
- C:/quantlib/QuantLib/ql/instruments/impliedvolatility.cpp

7.111 QuantLib::MakeCapFloor Class Reference

helper class

`#include <makecapfloor.hpp>`

Public Member Functions

- **MakeCapFloor** (CapFloor::Type capFloorType, const Period &capFloorTenor, const boost::shared_ptr< IborIndex > &iborIndex, Rate strike=Null< Rate >(), const Period &forwardStart=0 *Days)
- **operator CapFloor** () const
- **operator boost::shared_ptr< CapFloor >** () const
- **MakeCapFloor** & **withNominal** (Real n)
- **MakeCapFloor** & **withEffectiveDate** (const Date &effectiveDate, bool firstCapletExcluded)
- **MakeCapFloor** & **withTenor** (const Period &t)
- **MakeCapFloor** & **withCalendar** (const Calendar &cal)
- **MakeCapFloor** & **withConvention** (BusinessDayConvention bdc)
- **MakeCapFloor** & **withTerminationDateConvention** (BusinessDayConvention bdc)
- **MakeCapFloor** & **withRule** (DateGeneration::Rule r)
- **MakeCapFloor** & **withEndOfMonth** (bool flag=true)
- **MakeCapFloor** & **withFirstDate** (const Date &d)
- **MakeCapFloor** & **withNextToLastDate** (const Date &d)
- **MakeCapFloor** & **withDayCount** (const DayCounter &dc)
- **MakeCapFloor** & **asOptionlet** (bool b=true)

  only get last coupon
- **MakeCapFloor** & **withPricingEngine** (const boost::shared_ptr< PricingEngine > &engine)

7.111.1 Detailed Description

helper class

This class provides a more comfortable way to instantiate standard market cap and floor.

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

- C:/quantlib/QuantLib/ql/instruments/**makecapfloor.hpp**
- C:/quantlib/QuantLib/ql/instruments/makecapfloor.cpp

7.112 QuantLib::MakeCms Class Reference

helper class for instantiating CMS

`#include <makecms.hpp>`

Public Member Functions

- **MakeCms** (const Period &swapTenor, const boost::shared_ptr< SwapIndex > &swapIndex, const boost::shared_ptr< IborIndex > &iborIndex, Spread iborSpread=0.0, const Period &forwardStart=0 *Days)
- **MakeCms** (const Period &swapTenor, const boost::shared_ptr< SwapIndex > &swapIndex, Spread iborSpread=0.0, const Period &forwardStart=0 *Days)
- **operator Swap** () const
- **operator boost::shared_ptr< Swap >** () const
- **MakeCms** & **receiveCms** (bool flag=true)
- **MakeCms** & **withNominal** (Real n)
- **MakeCms** & **withEffectiveDate** (const Date &)
- **MakeCms** & **withCmsLegTenor** (const Period &t)
- **MakeCms** & **withCmsLegCalendar** (const Calendar &cal)
- **MakeCms** & **withCmsLegConvention** (BusinessDayConvention bdc)
- **MakeCms** & **withCmsLegTerminationDateConvention** (BusinessDayConvention)
- **MakeCms** & **withCmsLegRule** (DateGeneration::Rule r)
- **MakeCms** & **withCmsLegEndOfMonth** (bool flag=true)
- **MakeCms** & **withCmsLegFirstDate** (const Date &d)
- **MakeCms** & **withCmsLegNextToLastDate** (const Date &d)
- **MakeCms** & **withCmsLegDayCount** (const DayCounter &dc)
- **MakeCms** & **withFloatingLegTenor** (const Period &t)
- **MakeCms** & **withFloatingLegCalendar** (const Calendar &cal)
- **MakeCms** & **withFloatingLegConvention** (BusinessDayConvention bdc)
- **MakeCms** & **withFloatingLegTerminationDateConvention** (BusinessDayConvention bdc)
- **MakeCms** & **withFloatingLegRule** (DateGeneration::Rule r)
- **MakeCms** & **withFloatingLegEndOfMonth** (bool flag=true)
- **MakeCms** & **withFloatingLegFirstDate** (const Date &d)
- **MakeCms** & **withFloatingLegNextToLastDate** (const Date &d)
- **MakeCms** & **withFloatingLegDayCount** (const DayCounter &dc)
- **MakeCms** & **withAtmSpread** (bool flag=true)
- **MakeCms** & **withDiscountingTermStructure** (const Handle< YieldTermStructure > &discountingTermStructure)
- **MakeCms** & **withCmsCouponPricer** (const boost::shared_ptr< CmsCouponPricer > &couponPricer)

7.112.1 Detailed Description

helper class for instantiating CMS

This class provides a more comfortable way to instantiate standard market constant maturity swap.

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

- C:/quantlib/QuantLib/ql/instruments/**makecms.hpp**
- C:/quantlib/QuantLib/ql/instruments/makecms.cpp

7.113 QuantLib::MakeOIS Class Reference

helper class

`#include <makeois.hpp>`

Public Member Functions

- **MakeOIS** (const Period &swapTenor, const boost::shared_ptr< OvernightIndex > &overnightIndex, Rate fixedRate=Null< Rate >(), const Period &fwdStart=0 *Days)
- **operator OvernightIndexedSwap** () const
- **operator boost::shared_ptr< OvernightIndexedSwap >** () const
- **MakeOIS** & **receiveFixed** (bool flag=true)
- **MakeOIS** & **withType** (OvernightIndexedSwap::Type type)
- **MakeOIS** & **withNominal** (Real n)
- **MakeOIS** & **withSettlementDays** (Natural settlementDays)
- **MakeOIS** & **withEffectiveDate** (const Date &)
- **MakeOIS** & **withTerminationDate** (const Date &)
- **MakeOIS** & **withRule** (DateGeneration::Rule r)
- **MakeOIS** & **withPaymentFrequency** (Frequency f)
- **MakeOIS** & **withEndOfMonth** (bool flag=true)
- **MakeOIS** & **withFixedLegDayCount** (const DayCounter &dc)
- **MakeOIS** & **withOvernightLegSpread** (Spread sp)
- **MakeOIS** & **withDiscountingTermStructure** (const Handle< YieldTermStructure > &discountingTerm↩ Structure)
- **MakeOIS** & **withPricingEngine** (const boost::shared_ptr< PricingEngine > &engine)

7.113.1 Detailed Description

helper class

This class provides a more comfortable way to instantiate overnight indexed swaps.

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

- C:/quantlib/QuantLib/ql/instruments/**makeois.hpp**
- C:/quantlib/QuantLib/ql/instruments/makeois.cpp

7.114 QuantLib::MakeSwaption Class Reference

helper class

`#include <makeswaption.hpp>`

Public Member Functions

- **MakeSwaption** (const boost::shared_ptr< SwapIndex > &swapIndex, const Period &optionTenor, Rate strike=Null< Rate >())
- **MakeSwaption** (const boost::shared_ptr< SwapIndex > &swapIndex, const Date &fixingDate, Rate strike=Null< Rate >())
- **operator Swaption** () const
- **operator boost::shared_ptr< Swaption >** () const
- **MakeSwaption** & **withSettlementType** (Settlement::Type delivery)
- **MakeSwaption** & **withOptionConvention** (BusinessDayConvention bdc)
- **MakeSwaption** & **withExerciseDate** (const Date &)
- **MakeSwaption** & **withUnderlyingType** (const VanillaSwap::Type type)
- **MakeSwaption** & **withPricingEngine** (const boost::shared_ptr< PricingEngine > &engine)

7.114.1 Detailed Description

helper class

This class provides a more comfortable way to instantiate standard market swaption.

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

- C:/quantlib/QuantLib/ql/instruments/**makeswaption.hpp**
- C:/quantlib/QuantLib/ql/instruments/makeswaption.cpp

7.115 QuantLib::MakeVanillaSwap Class Reference

helper class

`#include <makevanillaswap.hpp>`

Public Member Functions

- **MakeVanillaSwap** (const Period &swapTenor, const boost::shared_ptr< IborIndex > &iborIndex, Rate fixedRate=Null< Rate >(), const Period &forwardStart=0 *Days)
- **operator VanillaSwap** () const
- **operator boost::shared_ptr< VanillaSwap >** () const
- **MakeVanillaSwap** & **receiveFixed** (bool flag=true)
- **MakeVanillaSwap** & **withType** (VanillaSwap::Type type)
- **MakeVanillaSwap** & **withNominal** (Real n)
- **MakeVanillaSwap** & **withSettlementDays** (Natural settlementDays)
- **MakeVanillaSwap** & **withEffectiveDate** (const Date &)
- **MakeVanillaSwap** & **withTerminationDate** (const Date &)
- **MakeVanillaSwap** & **withRule** (DateGeneration::Rule r)
- **MakeVanillaSwap** & **withFixedLegTenor** (const Period &t)
- **MakeVanillaSwap** & **withFixedLegCalendar** (const Calendar &cal)
- **MakeVanillaSwap** & **withFixedLegConvention** (BusinessDayConvention bdc)
- **MakeVanillaSwap** & **withFixedLegTerminationDateConvention** (BusinessDayConvention bdc)
- **MakeVanillaSwap** & **withFixedLegRule** (DateGeneration::Rule r)
- **MakeVanillaSwap** & **withFixedLegEndOfMonth** (bool flag=true)

- **MakeVanillaSwap** & **withFixedLegFirstDate** (const Date &d)
- **MakeVanillaSwap** & **withFixedLegNextToLastDate** (const Date &d)
- **MakeVanillaSwap** & **withFixedLegDayCount** (const DayCounter &dc)
- **MakeVanillaSwap** & **withFloatingLegTenor** (const Period &t)
- **MakeVanillaSwap** & **withFloatingLegCalendar** (const Calendar &cal)
- **MakeVanillaSwap** & **withFloatingLegConvention** (BusinessDayConvention bdc)
- **MakeVanillaSwap** & **withFloatingLegTerminationDateConvention** (BusinessDayConvention bdc)
- **MakeVanillaSwap** & **withFloatingLegRule** (DateGeneration::Rule r)
- **MakeVanillaSwap** & **withFloatingLegEndOfMonth** (bool flag=true)
- **MakeVanillaSwap** & **withFloatingLegFirstDate** (const Date &d)
- **MakeVanillaSwap** & **withFloatingLegNextToLastDate** (const Date &d)
- **MakeVanillaSwap** & **withFloatingLegDayCount** (const DayCounter &dc)
- **MakeVanillaSwap** & **withFloatingLegSpread** (Spread sp)
- **MakeVanillaSwap** & **withDiscountingTermStructure** (const Handle< YieldTermStructure > &discount↩ Curve)
- **MakeVanillaSwap** & **withPricingEngine** (const boost::shared_ptr< PricingEngine > &engine)

7.115.1 Detailed Description

helper class

This class provides a more comfortable way to instantiate standard market swap.

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

- C:/quantlib/QuantLib/ql/instruments/**makevanillaswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/makevanillaswap.cpp

7.116 QuantLib::MakeYoYInflationCapFloor Class Reference

helper class

`#include <makeyoyinflationcapfloor.hpp>`

Public Member Functions

- **MakeYoYInflationCapFloor** (YoYInflationCapFloor::Type capFloorType, const Size &length, const Calendar &cal, const boost::shared_ptr< YoYInflationIndex > &index, const Period &observationLag, Rate strike=Null< Rate >(), const Period &forwardStart=0 *Days)
- **MakeYoYInflationCapFloor** & **withNominal** (Real n)
- **MakeYoYInflationCapFloor** & **withEffectiveDate** (const Date &effectiveDate)
- **MakeYoYInflationCapFloor** & **withFirstCapletExcluded** ()
- **MakeYoYInflationCapFloor** & **withPaymentDayCounter** (const DayCounter &)
- **MakeYoYInflationCapFloor** & **withPaymentAdjustment** (BusinessDayConvention)
- **MakeYoYInflationCapFloor** & **withFixingDays** (Natural fixingDays)
- **operator YoYInflationCapFloor** () const
- **operator boost::shared_ptr< YoYInflationCapFloor >** () const
- **MakeYoYInflationCapFloor** & **asOptionlet** (bool b=true)

  only get last coupon
- **MakeYoYInflationCapFloor** & **withPricingEngine** (const boost::shared_ptr< PricingEngine > &engine)

7.116.1 Detailed Description

helper class

This class provides a more comfortable way to instantiate standard yoy inflation cap and floor.

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

- C:/quantlib/QuantLib/ql/instruments/makeyoyinflationcapfloor.hpp
- C:/quantlib/QuantLib/ql/instruments/makeyoyinflationcapfloor.cpp

7.117 QuantLib::MaxBasketPayoff Class Reference

Inheritance diagram for QuantLib::MaxBasketPayoff:

Collaboration diagram for QuantLib::MaxBasketPayoff:

Public Member Functions

- **MaxBasketPayoff** (const boost::shared_ptr< Payoff > &p)
- Real **accumulate** (const Array &a) const

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

- C:/quantlib/QuantLib/ql/instruments/**basketoption.hpp**

7.118 QuantLib::MinBasketPayoff Class Reference

Inheritance diagram for QuantLib::MinBasketPayoff:

Collaboration diagram for QuantLib::MinBasketPayoff:

Public Member Functions

- **MinBasketPayoff** (const boost::shared_ptr< Payoff > &p)
- Real **accumulate** (const Array &a) const

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

- C:/quantlib/QuantLib/ql/instruments/**basketoption.hpp**

7.119 QuantLib::MultiAssetOption Class Reference

Base class for options on multiple assets.

`#include <multiassetoption.hpp>`

Inheritance diagram for QuantLib::MultiAssetOption:

Collaboration diagram for QuantLib::MultiAssetOption:

Classes

- class **engine**
- class **results**

  Results from multi-asset option calculation

Public Member Functions

- **MultiAssetOption** (const boost::shared_ptr< Payoff > &, const boost::shared_ptr< Exercise > &)
- void **setupArguments** (PricingEngine::arguments *) const
- void **fetchResults** (const PricingEngine::results *) const

Instrument interface

- bool **isExpired** () const

greeks

- Real **delta** () const
- Real **gamma** () const
- Real **theta** () const
- Real **vega** () const
- Real **rho** () const
- Real **dividendRho** () const

Protected Member Functions

- void **setupExpired** () const

Protected Attributes

- Real **delta_**
- Real **gamma_**
- Real **theta_**
- Real **vega_**
- Real **rho_**
- Real **dividendRho_**

7.119.1 Detailed Description

Base class for options on multiple assets.

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

- C:/quantlib/QuantLib/ql/instruments/**multiassetoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/multiassetoption.cpp

7.120 QuantLib::NonstandardSwap Class Reference

nonstandard swap

`#include <nonstandardswap.hpp>`

Inheritance diagram for QuantLib::NonstandardSwap:

Collaboration diagram for QuantLib::NonstandardSwap:

Classes

- class **arguments**

  Arguments for nonstandard swap calculation
- class **engine**
- class **results**

  Results from nonstandard swap calculation

Public Member Functions

- **NonstandardSwap** (const **VanillaSwap** &fromVanilla)
- **NonstandardSwap** (const VanillaSwap::Type type, const std::vector< Real > &fixedNominal, const std::vector< Real > &floatingNominal, const Schedule &fixedSchedule, const std::vector< Real > &fixedRate, const DayCounter &fixedDayCount, const Schedule &floatingSchedule, const boost::shared_ptr< IborIndex > &iborIndex, const Real gearing, const Spread spread, const DayCounter &floatingDayCount, const bool intermediateCapitalExchange=false, const bool finalCapitalExchange=false, boost::optional< BusinessDayConvention > paymentConvention=boost::none)
- void **setupArguments** (PricingEngine::arguments *args) const
- void **fetchResults** (const PricingEngine::results *) const

Inspectors

- VanillaSwap::Type **type** () const
- const std::vector< Real > & **fixedNominal** () const
- const std::vector< Real > & **floatingNominal** () const
- const Schedule & **fixedSchedule** () const
- const std::vector< Real > & **fixedRate** () const
- const DayCounter & **fixedDayCount** () const
- const Schedule & **floatingSchedule** () const
- const boost::shared_ptr< IborIndex > & **iborIndex** () const
- const Spread **spread** () const
- const Real **gearing** () const
- const DayCounter & **floatingDayCount** () const
- BusinessDayConvention **paymentConvention** () const
- const Leg & **fixedLeg** () const
- const Leg & **floatingLeg** () const

Additional Inherited Members

7.120.1 Detailed Description

nonstandard swap

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

- C:/quantlib/QuantLib/ql/instruments/**nonstandardswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/nonstandardswap.cpp

7.121 QuantLib::NonstandardSwaption Class Reference

nonstandard swaption class

`#include <nonstandardswaption.hpp>`

Inheritance diagram for QuantLib::NonstandardSwaption:

Collaboration diagram for QuantLib::NonstandardSwaption:

Classes

- class **arguments**

  Arguments for nonstandard swaption calculation
- class **engine**

  base class for nonstandard swaption engines

Public Member Functions

- **NonstandardSwaption** (const **Swaption** &fromSwaption)
- **NonstandardSwaption** (const boost::shared_ptr< **NonstandardSwap** > &swap, const boost::shared_ptr< Exercise > &exercise, Settlement::Type delivery=Settlement::Physical)
- Disposable< std::vector< boost::shared_ptr< CalibrationHelper > > > **calibrationBasket** (boost::shared_ptr< SwapIndex > standardSwapBase, boost::shared_ptr< SwaptionVolatilityStructure > swaptionVolatility, const BasketGeneratingEngine::CalibrationBasketType basketType=BasketGeneratingEngine::MaturityStrikeByDeltaGamma) const

Instrument interface

- bool **isExpired** () const
- void **setupArguments** (PricingEngine::arguments *) const

Inspectors

- VanillaSwap::Type **type** () const
- const boost::shared_ptr< **NonstandardSwap** > & **underlyingSwap** () const

7.121.1 Detailed Description

nonstandard swaption class

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

- C:/quantlib/QuantLib/ql/instruments/**nonstandardswaption.hpp**
- C:/quantlib/QuantLib/ql/instruments/nonstandardswaption.cpp

7.122 QuantLib::NullPayoff Class Reference

Dummy payoff class.

`#include <payoffs.hpp>`

Inheritance diagram for QuantLib::NullPayoff:

Collaboration diagram for QuantLib::NullPayoff:

Public Member Functions

Payoff interface

- std::string **name** () const
- std::string **description** () const
- Real **operator()** (Real price) const
- virtual void **accept** (AcyclicVisitor &)

7.122.1 Detailed Description

Dummy payoff class.

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

- C:/quantlib/QuantLib/ql/instruments/**payoffs.hpp**
- C:/quantlib/QuantLib/ql/instruments/payoffs.cpp

7.123 QuantLib::OneAssetOption Class Reference

Base class for options on a single asset.

`#include <oneassetoption.hpp>`

Inheritance diagram for QuantLib::OneAssetOption:

Collaboration diagram for QuantLib::OneAssetOption:

Classes

- class **engine**
- class **results**

  Results from single-asset option calculation

Public Member Functions

- **OneAssetOption** (const boost::shared_ptr< Payoff > &, const boost::shared_ptr< Exercise > &)
- void **fetchResults** (const PricingEngine::results *) const

Instrument interface

- bool **isExpired** () const

greeks

- Real **delta** () const
- Real **deltaForward** () const
- Real **elasticity** () const
- Real **gamma** () const
- Real **theta** () const
- Real **thetaPerDay** () const
- Real **vega** () const
- Real **rho** () const
- Real **dividendRho** () const
- Real **strikeSensitivity** () const
- Real **itmCashProbability** () const

Protected Member Functions

- void **setupExpired** () const

Protected Attributes

- Real **delta_**
- Real **deltaForward_**
- Real **elasticity_**
- Real **gamma_**
- Real **theta_**
- Real **thetaPerDay_**
- Real **vega_**
- Real **rho_**
- Real **dividendRho_**
- Real **strikeSensitivity_**
- Real **itmCashProbability_**

7.123.1 Detailed Description

Base class for options on a single asset.

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

- C:/quantlib/QuantLib/ql/instruments/**oneassetoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/oneassetoption.cpp

7.124 QuantLib::OvernightIndexedSwap Class Reference

Overnight indexed swap: fix vs compounded overnight rate.

`#include <overnightindexedswap.hpp>`

Inheritance diagram for QuantLib::OvernightIndexedSwap:

Collaboration diagram for QuantLib::OvernightIndexedSwap:

Public Types

- enum **Type** { **Receiver** = -1, **Payer** = 1 }

Public Member Functions

- **OvernightIndexedSwap** (Type type, Real nominal, const Schedule &schedule, Rate fixedRate, const DayCounter &fixedDC, const boost::shared_ptr< OvernightIndex > &overnightIndex, Spread spread=0.0)
- **OvernightIndexedSwap** (Type type, std::vector< Real > nominals, const Schedule &schedule, Rate fixedRate, const DayCounter &fixedDC, const boost::shared_ptr< OvernightIndex > &overnightIndex, Spread spread=0.0)

Inspectors

- Type **type** () const
- Real **nominal** () const
- std::vector< Real > **nominals** () const
- Frequency **paymentFrequency** ()
- Rate **fixedRate** () const
- const DayCounter & **fixedDayCount** ()
- const boost::shared_ptr< OvernightIndex > & **overnightIndex** ()
- Spread **spread** ()
- const Leg & **fixedLeg** () const
- const Leg & **overnightLeg** () const

Results

- Real **fixedLegBPS** () const
- Real **fixedLegNPV** () const
- Real **fairRate** () const
- Real **overnightLegBPS** () const
- Real **overnightLegNPV** () const
- Spread **fairSpread** () const

Additional Inherited Members

7.124.1 Detailed Description

Overnight indexed swap: fix vs compounded overnight rate.

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

- C:/quantlib/QuantLib/ql/instruments/**overnightindexedswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/overnightindexedswap.cpp

7.125 QuantLib::PercentageStrikePayoff Class Reference

Payoff with strike expressed as percentage

`#include <payoffs.hpp>`

Inheritance diagram for QuantLib::PercentageStrikePayoff:

Collaboration diagram for QuantLib::PercentageStrikePayoff:

Public Member Functions

- **PercentageStrikePayoff** (Option::Type type, Real moneyness)

Payoff interface

- std::string **name** () const
- Real **operator()** (Real price) const
- virtual void **accept** (AcyclicVisitor &)

Additional Inherited Members

7.125.1 Detailed Description

Payoff with strike expressed as percentage

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

- C:/quantlib/QuantLib/ql/instruments/**payoffs.hpp**
- C:/quantlib/QuantLib/ql/instruments/payoffs.cpp

7.126 QuantLib::PlainVanillaPayoff Class Reference

Plain-vanilla payoff.

`#include <payoffs.hpp>`

Inheritance diagram for QuantLib::PlainVanillaPayoff:

Collaboration diagram for QuantLib::PlainVanillaPayoff:

Public Member Functions

- **PlainVanillaPayoff** (Option::Type type, Real strike)

Payoff interface

- std::string **name** () const
- Real **operator()** (Real price) const
- virtual void **accept** (AcyclicVisitor &)

Additional Inherited Members

7.126.1 Detailed Description

Plain-vanilla payoff.

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

- C:/quantlib/QuantLib/ql/instruments/**payoffs.hpp**
- C:/quantlib/QuantLib/ql/instruments/payoffs.cpp

7.127 QuantLib::Callability::Price Class Reference

amount to be paid upon callability

`#include <callabilityschedule.hpp>`

Public Types

- enum **Type** { **Dirty**, **Clean** }

Public Member Functions

- **Price** (Real amount, Type type)
- Real **amount** () const
- Type **type** () const

7.127.1 Detailed Description

amount to be paid upon callability

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

- C:/quantlib/QuantLib/ql/instruments/**callabilityschedule.hpp**

7.128 QuantLib::QuantoBarrierOption Class Reference

Quanto version of a barrier option.

`#include <quantobarrieroption.hpp>`

Inheritance diagram for QuantLib::QuantoBarrierOption:

Collaboration diagram for QuantLib::QuantoBarrierOption:

Public Types

- typedef **BarrierOption::arguments arguments**
- typedef **QuantoOptionResults**< **BarrierOption::results** > **results**

Public Member Functions

- **QuantoBarrierOption** (Barrier::Type barrierType, Real barrier, Real rebate, const boost::shared_ptr< **StrikedTypePayoff** > &payoff, const boost::shared_ptr< Exercise > &exercise)
- void **fetchResults** (const PricingEngine::results *) const

greeks

- Real **qvega** () const
- Real **qrho** () const
- Real **qlambda** () const

Additional Inherited Members

7.128.1 Detailed Description

Quanto version of a barrier option.

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

- C:/quantlib/QuantLib/ql/instruments/**quantobarrieroption.hpp**
- C:/quantlib/QuantLib/ql/instruments/quantobarrieroption.cpp

7.129 QuantLib::QuantoForwardVanillaOption Class Reference

Quanto version of a forward vanilla option.

`#include <quantoforwardvanillaoption.hpp>`

Inheritance diagram for QuantLib::QuantoForwardVanillaOption:

Collaboration diagram for QuantLib::QuantoForwardVanillaOption:

Public Types

- typedef **ForwardVanillaOption::arguments arguments**
- typedef **QuantoOptionResults**< **ForwardVanillaOption::results** > **results**

Public Member Functions

- **QuantoForwardVanillaOption** (Real moneyness, const Date &resetDate, const boost::shared_ptr< **StrikedTypePayoff** > &, const boost::shared_ptr< Exercise > &)
- void **fetchResults** (const PricingEngine::results *) const

greeks

- Real **qvega** () const
- Real **qrho** () const
- Real **qlambda** () const

Additional Inherited Members

7.129.1 Detailed Description

Quanto version of a forward vanilla option.

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

- C:/quantlib/QuantLib/ql/instruments/**quantoforwardvanillaoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/quantoforwardvanillaoption.cpp

7.130 QuantLib::QuantoOptionResults< ResultsType > Class Template Reference

Results from quanto option calculation

`#include <quantovanillaoption.hpp>`

Inheritance diagram for QuantLib::QuantoOptionResults< ResultsType >:

Collaboration diagram for QuantLib::QuantoOptionResults< ResultsType >:

Public Member Functions

- void **reset** ()

Public Attributes

- Real **qvega**
- Real **qrho**
- Real **qlambda**

7.130.1 Detailed Description

template<class ResultsType>
class QuantLib::QuantoOptionResults< ResultsType >

Results from quanto option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**quantovanillaoption.hpp**

7.131 QuantLib::QuantoVanillaOption Class Reference

quanto version of a vanilla option

`#include <quantovanillaoption.hpp>`

Inheritance diagram for QuantLib::QuantoVanillaOption:

Collaboration diagram for QuantLib::QuantoVanillaOption:

Public Types

- typedef OneAssetOption::arguments **arguments**
- typedef **QuantoOptionResults**< **OneAssetOption::results** > **results**
- typedef GenericEngine< arguments, **results** > **engine**

Public Member Functions

- **QuantoVanillaOption** (const boost::shared_ptr< **StrikedTypePayoff** > &, const boost::shared_ptr< Exercise > &)
- void **fetchResults** (const PricingEngine::results *) const

greeks

- Real **qvega** () const
- Real **qrho** () const
- Real **qlambda** () const

Additional Inherited Members

7.131.1 Detailed Description

quanto version of a vanilla option

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

- C:/quantlib/QuantLib/ql/instruments/**quantovanillaoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/quantovanillaoption.cpp

7.132 QuantLib::RatchetMaxPayoff Class Reference

RatchetMax payoff (double option)

`#include <stickyratchet.hpp>`

Inheritance diagram for QuantLib::RatchetMaxPayoff:

Collaboration diagram for QuantLib::RatchetMaxPayoff:

Public Member Functions

- **RatchetMaxPayoff** (Real gearing1, Real gearing2, Real gearing3, Real spread1, Real spread2, Real spread3, Real initialValue1, Real initialValue2, Real accrualFactor)

Payoff interface

- std::string **name** () const

Additional Inherited Members

7.132.1 Detailed Description

RatchetMax payoff (double option)

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

- C:/quantlib/QuantLib/ql/instruments/**stickyratchet.hpp**

7.133 QuantLib::RatchetMinPayoff Class Reference

RatchetMin payoff (double option)

`#include <stickyratchet.hpp>`

Inheritance diagram for QuantLib::RatchetMinPayoff:

Collaboration diagram for QuantLib::RatchetMinPayoff:

Public Member Functions

- **RatchetMinPayoff** (Real gearing1, Real gearing2, Real gearing3, Real spread1, Real spread2, Real spread3, Real initialValue1, Real initialValue2, Real accrualFactor)

Payoff interface

- std::string **name** () const

Additional Inherited Members

7.133.1 Detailed Description

RatchetMin payoff (double option)

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

- C:/quantlib/QuantLib/ql/instruments/**stickyratchet.hpp**

7.134 QuantLib::RatchetPayoff Class Reference

Ratchet payoff (single option)

`#include <stickyratchet.hpp>`

Inheritance diagram for QuantLib::RatchetPayoff:

Collaboration diagram for QuantLib::RatchetPayoff:

Public Member Functions

- **RatchetPayoff** (Real gearing1, Real gearing2, Real spread1, Real spread2, Real initialValue, Real accrual↩ Factor)

Payoff interface

- std::string **name** () const

Additional Inherited Members

7.134.1 Detailed Description

Ratchet payoff (single option)

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

- C:/quantlib/QuantLib/ql/instruments/**stickyratchet.hpp**

7.135 QuantLib::RendistatoBasket Class Reference

Inheritance diagram for QuantLib::RendistatoBasket:

Collaboration diagram for QuantLib::RendistatoBasket:

Public Member Functions

- **RendistatoBasket** (const std::vector< boost::shared_ptr< **BTP** > > &btps, const std::vector< Real > &outstandings, const std::vector< Handle< Quote > > &cleanPriceQuotes)

Inspectors

- Size **size** () const
- const std::vector< boost::shared_ptr< **BTP** > > & **btps** () const
- const std::vector< Handle< Quote > > & **cleanPriceQuotes** () const
- const std::vector< Real > & **outstandings** () const
- const std::vector< Real > & **weights** () const
- Real **outstanding** () const

Observer interface

- void **update** ()

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

- C:/quantlib/QuantLib/ql/instruments/bonds/**btp.hpp**
- C:/quantlib/QuantLib/ql/instruments/bonds/btp.cpp

7.136 QuantLib::RendistatoCalculator Class Reference

Inheritance diagram for QuantLib::RendistatoCalculator:

Collaboration diagram for QuantLib::RendistatoCalculator:

Public Member Functions

- **RendistatoCalculator** (const boost::shared_ptr< **RendistatoBasket** > &basket, const boost::shared_ptr< Euribor > &euriborIndex, const Handle< YieldTermStructure > &discountCurve)

Calculations

- Rate **yield** () const
- Time **duration** () const
- const std::vector< Rate > & **yields** () const
- const std::vector< Time > & **durations** () const
- const std::vector< Time > & **swapLengths** () const
- const std::vector< Rate > & **swapRates** () const
- const std::vector< Rate > & **swapYields** () const
- const std::vector< Time > & **swapDurations** () const

Equivalent Swap proxy

- boost::shared_ptr< **VanillaSwap** > **equivalentSwap** () const
- Rate **equivalentSwapRate** () const
- Rate **equivalentSwapYield** () const
- Time **equivalentSwapDuration** () const
- Time **equivalentSwapLength** () const
- Spread **equivalentSwapSpread** () const

Protected Member Functions

LazyObject interface

- void **performCalculations** () const

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

- C:/quantlib/QuantLib/ql/instruments/bonds/**btp.hpp**
- C:/quantlib/QuantLib/ql/instruments/bonds/btp.cpp

7.137 QuantLib::RendistatoEquivalentSwapLengthQuote Class Reference

RendistatoCalculator (p. 122) equivalent swap lenth Quote adapter.

`#include <btp.hpp>`

Inheritance diagram for QuantLib::RendistatoEquivalentSwapLengthQuote:

Collaboration diagram for QuantLib::RendistatoEquivalentSwapLengthQuote:

Public Member Functions

- **RendistatoEquivalentSwapLengthQuote** (const boost::shared_ptr< **RendistatoCalculator** > &r)
- Real **value** () const
- bool **isValid** () const

7.137.1 Detailed Description

RendistatoCalculator (p. 122) equivalent swap lenth Quote adapter.

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

- C:/quantlib/QuantLib/ql/instruments/bonds/**btp.hpp**
- C:/quantlib/QuantLib/ql/instruments/bonds/btp.cpp

7.138 QuantLib::RendistatoEquivalentSwapSpreadQuote Class Reference

RendistatoCalculator (p. 122) equivalent swap spread Quote adapter.

`#include <btp.hpp>`

Inheritance diagram for QuantLib::RendistatoEquivalentSwapSpreadQuote:

Collaboration diagram for QuantLib::RendistatoEquivalentSwapSpreadQuote:

Public Member Functions

- **RendistatoEquivalentSwapSpreadQuote** (const boost::shared_ptr< **RendistatoCalculator** > &r)
- Real **value** () const
- bool **isValid** () const

7.138.1 Detailed Description

RendistatoCalculator (p. 122) equivalent swap spread Quote adapter.

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

- C:/quantlib/QuantLib/ql/instruments/bonds/**btp.hpp**
- C:/quantlib/QuantLib/ql/instruments/bonds/btp.cpp

7.139 QuantLib::CPICapFloor::results Class Reference

Inheritance diagram for QuantLib::CPICapFloor::results:

Collaboration diagram for QuantLib::CPICapFloor::results:

Public Member Functions

- void **reset** ()

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

- C:/quantlib/QuantLib/ql/instruments/**cpicapfloor.hpp**
- C:/quantlib/QuantLib/ql/instruments/cpicapfloor.cpp

7.140 QuantLib::CreditDefaultSwap::results Class Reference

Inheritance diagram for QuantLib::CreditDefaultSwap::results:

Collaboration diagram for QuantLib::CreditDefaultSwap::results:

Public Member Functions

- void **reset** ()

Public Attributes

- Rate **fairSpread**
- Rate **fairUpfront**
- Real **couponLegBPS**
- Real **couponLegNPV**
- Real **defaultLegNPV**
- Real **upfrontBPS**
- Real **upfrontNPV**

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

- C:/quantlib/QuantLib/ql/instruments/**creditdefaultswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/creditdefaultswap.cpp

7.141 QuantLib::VanillaSwap::results Class Reference

Results from simple swap calculation

`#include <vanillaswap.hpp>`

Inheritance diagram for QuantLib::VanillaSwap::results:

Collaboration diagram for QuantLib::VanillaSwap::results:

Public Member Functions

- void **reset** ()

Public Attributes

- Rate **fairRate**
- Spread **fairSpread**

7.141.1 Detailed Description

Results from simple swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**vanillaswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/vanillaswap.cpp

7.142 QuantLib::Swap::results Class Reference

Inheritance diagram for QuantLib::Swap::results:

Collaboration diagram for QuantLib::Swap::results:

Public Member Functions

- void **reset** ()

Public Attributes

- std::vector< Real > **legNPV**
- std::vector< Real > **legBPS**
- std::vector< DiscountFactor > **startDiscounts**
- std::vector< DiscountFactor > **endDiscounts**
- DiscountFactor **npvDateDiscount**

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

- C:/quantlib/QuantLib/ql/instruments/**swap.hpp**
- C:/quantlib/QuantLib/ql/instruments/swap.cpp

7.143 QuantLib::OneAssetOption::results Class Reference

Results from single-asset option calculation

`#include <oneassetoption.hpp>`

Inheritance diagram for QuantLib::OneAssetOption::results:

Collaboration diagram for QuantLib::OneAssetOption::results:

Public Member Functions

- void **reset** ()

7.143.1 Detailed Description

Results from single-asset option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**oneassetoption.hpp**

7.144 QuantLib::CPISwap::results Class Reference

Results from swap calculation

`#include <cpiswap.hpp>`

Inheritance diagram for QuantLib::CPISwap::results:

Collaboration diagram for QuantLib::CPISwap::results:

Public Member Functions

- void **reset** ()

Public Attributes

- Rate **fairRate**
- Spread **fairSpread**

7.144.1 Detailed Description

Results from swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**cpiswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/cpiswap.cpp

7.145 QuantLib::FloatFloatSwap::results Class Reference

Results from float float swap calculation

```
#include <floatfloatswap.hpp>
```

Inheritance diagram for QuantLib::FloatFloatSwap::results:

Collaboration diagram for QuantLib::FloatFloatSwap::results:

Public Member Functions

- void **reset** ()

Additional Inherited Members

7.145.1 Detailed Description

Results from float float swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**floatfloatswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/floatfloatswap.cpp

7.146 QuantLib::NonstandardSwap::results Class Reference

Results from nonstandard swap calculation

`#include <nonstandardswap.hpp>`

Inheritance diagram for QuantLib::NonstandardSwap::results:

Collaboration diagram for QuantLib::NonstandardSwap::results:

Public Member Functions

- void **reset** ()

Additional Inherited Members

7.146.1 Detailed Description

Results from nonstandard swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**nonstandardswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/nonstandardswap.cpp

7.147 QuantLib::AssetSwap::results Class Reference

Results from simple swap calculation

`#include <assetswap.hpp>`

Inheritance diagram for QuantLib::AssetSwap::results:

Collaboration diagram for QuantLib::AssetSwap::results:

Public Member Functions

- void **reset** ()

Public Attributes

- Spread **fairSpread**
- Real **fairCleanPrice**
- Real **fairNonParRepayment**

7.147.1 Detailed Description

Results from simple swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**assetswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/assetswap.cpp

7.148 QuantLib::Bond::results Class Reference

Inheritance diagram for QuantLib::Bond::results:

Collaboration diagram for QuantLib::Bond::results:

Public Member Functions

- void **reset** ()

Public Attributes

- Real **settlementValue**

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

- C:/quantlib/QuantLib/ql/instruments/**bond.hpp**

7.149 QuantLib::VarianceSwap::results Class Reference

Results from variance-swap calculation

```
#include <varianceswap.hpp>
```

Inheritance diagram for QuantLib::VarianceSwap::results:

Collaboration diagram for QuantLib::VarianceSwap::results:

Public Member Functions

- void **reset** ()

Public Attributes

- Real **variance**

7.149.1 Detailed Description

Results from variance-swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**varianceswap.hpp**

7.150 QuantLib::YearOnYearInflationSwap::results Class Reference

Results from YoY swap calculation

`#include <yearonyearinflationswap.hpp>`

Inheritance diagram for QuantLib::YearOnYearInflationSwap::results:

Collaboration diagram for QuantLib::YearOnYearInflationSwap::results:

Public Member Functions

- void **reset** ()

Public Attributes

- Rate **fairRate**
- Spread **fairSpread**

7.150.1 Detailed Description

Results from YoY swap calculation

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

- C:/quantlib/QuantLib/ql/instruments/**yearonyearinflationswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/yearonyearinflationswap.cpp

7.151 QuantLib::MultiAssetOption::results Class Reference

Results from multi-asset option calculation

`#include <multiassetoption.hpp>`

Inheritance diagram for QuantLib::MultiAssetOption::results:

Collaboration diagram for QuantLib::MultiAssetOption::results:

Public Member Functions

- void **reset** ()

7.151.1 Detailed Description

Results from multi-asset option calculation

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

- C:/quantlib/QuantLib/ql/instruments/**multiassetoption.hpp**

7.152 QuantLib::Settlement Struct Reference

settlement information

`#include <swaption.hpp>`

Public Types

- enum **Type** { **Physical**, **Cash** }

7.152.1 Detailed Description

settlement information

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

- C:/quantlib/QuantLib/ql/instruments/**swaption.hpp**

7.153 QuantLib::SpreadBasketPayoff Class Reference

Inheritance diagram for QuantLib::SpreadBasketPayoff:

Collaboration diagram for QuantLib::SpreadBasketPayoff:

Public Member Functions

- **SpreadBasketPayoff** (const boost::shared_ptr< Payoff > &p)
- Real **accumulate** (const Array &a) const

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

- C:/quantlib/QuantLib/ql/instruments/**basketoption.hpp**

7.154 QuantLib::StickyMaxPayoff Class Reference

StickyMax payoff (double option)

`#include <stickyratchet.hpp>`

Inheritance diagram for QuantLib::StickyMaxPayoff:

Collaboration diagram for QuantLib::StickyMaxPayoff:

Public Member Functions

- **StickyMaxPayoff** (Real gearing1, Real gearing2, Real gearing3, Real spread1, Real spread2, Real spread3, Real initialValue1, Real initialValue2, Real accrualFactor)

Payoff interface

- std::string **name** () const

Additional Inherited Members

7.154.1 Detailed Description

StickyMax payoff (double option)

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

- C:/quantlib/QuantLib/ql/instruments/**stickyratchet.hpp**

7.155 QuantLib::StickyMinPayoff Class Reference

StickyMin payoff (double option)

`#include <stickyratchet.hpp>`

Inheritance diagram for QuantLib::StickyMinPayoff:

Collaboration diagram for QuantLib::StickyMinPayoff:

Public Member Functions

- **StickyMinPayoff** (Real gearing1, Real gearing2, Real gearing3, Real spread1, Real spread2, Real spread3, Real initialValue1, Real initialValue2, Real accrualFactor)

Payoff interface

- std::string **name** () const

Additional Inherited Members

7.155.1 Detailed Description

StickyMin payoff (double option)

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

- C:/quantlib/QuantLib/ql/instruments/**stickyratchet.hpp**

7.156 QuantLib::StickyPayoff Class Reference

Sticky payoff (single option)

`#include <stickyratchet.hpp>`

Inheritance diagram for QuantLib::StickyPayoff:

Collaboration diagram for QuantLib::StickyPayoff:

Public Member Functions

- **StickyPayoff** (Real gearing1, Real gearing2, Real spread1, Real spread2, Real initialValue, Real accrual↩ Factor)

Payoff interface

- std::string **name** () const

Additional Inherited Members

7.156.1 Detailed Description

Sticky payoff (single option)

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

- C:/quantlib/QuantLib/ql/instruments/**stickyratchet.hpp**

7.157 QuantLib::Stock Class Reference

Simple stock class.

`#include <stock.hpp>`

Inheritance diagram for QuantLib::Stock:

Collaboration diagram for QuantLib::Stock:

Public Member Functions

- **Stock** (const Handle< Quote > &quote)
- bool **isExpired** () const

Protected Member Functions

- void **performCalculations** () const

7.157.1 Detailed Description

Simple stock class.

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

- C:/quantlib/QuantLib/ql/instruments/**stock.hpp**
- C:/quantlib/QuantLib/ql/instruments/stock.cpp

7.158 QuantLib::StrikedTypePayoff Class Reference

Intermediate class for payoffs based on a fixed strike.

`#include <payoffs.hpp>`

Inheritance diagram for QuantLib::StrikedTypePayoff:

Collaboration diagram for QuantLib::StrikedTypePayoff:

Public Member Functions

- Real **strike** () const

Payoff interface

- std::string **description** () const

Protected Member Functions

- **StrikedTypePayoff** (Option::Type type, Real strike)

Protected Attributes

- Real **strike_**

7.158.1 Detailed Description

Intermediate class for payoffs based on a fixed strike.

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

- C:/quantlib/QuantLib/ql/instruments/**payoffs.hpp**
- C:/quantlib/QuantLib/ql/instruments/payoffs.cpp

7.159 QuantLib::SuperFundPayoff Class Reference

Binary supershare and superfund payoffs.

`#include <payoffs.hpp>`

Inheritance diagram for QuantLib::SuperFundPayoff:

Collaboration diagram for QuantLib::SuperFundPayoff:

Public Member Functions

- **SuperFundPayoff** (Real strike, Real secondStrike)
- Real **secondStrike** () const

Payoff interface

- std::string **name** () const
- Real **operator()** (Real price) const
- virtual void **accept** (AcyclicVisitor &)

Protected Attributes

- Real **secondStrike_**

Additional Inherited Members

7.159.1 Detailed Description

Binary supershare and superfund payoffs.

Binary superfund payoff

Superfund sometimes also called "supershare", which can lead to ambiguity; within QuantLib the terms supershare and superfund are used consistently according to the definitions in Bloomberg OVX function's help pages.

This payoff is equivalent to being (1/lowerstrike) a) long (short) an AssetOrNothing Call (Put) at the lower strike and b) short (long) an AssetOrNothing Call (Put) at the higher strike

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

- C:/quantlib/QuantLib/ql/instruments/**payoffs.hpp**
- C:/quantlib/QuantLib/ql/instruments/payoffs.cpp

7.160 QuantLib::SuperSharePayoff Class Reference

Binary supershare payoff.

`#include <payoffs.hpp>`

Inheritance diagram for QuantLib::SuperSharePayoff:

Collaboration diagram for QuantLib::SuperSharePayoff:

Public Member Functions

- **SuperSharePayoff** (Real strike, Real secondStrike, Real cashPayoff)
- Real **secondStrike** () const
- Real **cashPayoff** () const

Payoff interface

- std::string **name** () const
- std::string **description** () const
- Real **operator()** (Real price) const
- virtual void **accept** (AcyclicVisitor &)

Protected Attributes

- Real **secondStrike_**
- Real **cashPayoff_**

Additional Inherited Members

7.160.1 Detailed Description

Binary supershare payoff.

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

- C:/quantlib/QuantLib/ql/instruments/**payoffs.hpp**
- C:/quantlib/QuantLib/ql/instruments/payoffs.cpp

7.161 QuantLib::Swap Class Reference

Interest rate swap.

`#include <swap.hpp>`

Inheritance diagram for QuantLib::Swap:

Collaboration diagram for QuantLib::Swap:

Classes

- class **arguments**
- class **engine**
- class **results**

Public Member Functions

Additional interface

- Date **startDate** () const
- Date **maturityDate** () const
- Real **legBPS** (Size j) const
- Real **legNPV** (Size j) const
- DiscountFactor **startDiscounts** (Size j) const
- DiscountFactor **endDiscounts** (Size j) const
- DiscountFactor **npvDateDiscount** () const
- const Leg & **leg** (Size j) const

Protected Attributes

- std::vector< Leg > **legs_**
- std::vector< Real > **payer_**
- std::vector< Real > **legNPV_**
- std::vector< Real > **legBPS_**
- std::vector< DiscountFactor > **startDiscounts_**
- std::vector< DiscountFactor > **endDiscounts_**
- DiscountFactor **npvDateDiscount_**

Constructors

- **Swap** (const Leg &firstLeg, const Leg &secondLeg)
- **Swap** (const std::vector< Leg > &legs, const std::vector< bool > &payer)
- **Swap** (Size legs)

Instrument interface

- bool **isExpired** () const
- void **setupArguments** (PricingEngine::arguments *) const
- void **fetchResults** (const PricingEngine::results *) const
- void **setupExpired** () const

7.161.1 Detailed Description

Interest rate swap.

The cash flows belonging to the first leg are paid; the ones belonging to the second leg are received.

7.161.2 Constructor & Destructor Documentation

7.161.2.1 QuantLib::Swap::Swap (const Leg & *firstLeg,* const Leg & *secondLeg*)

The cash flows belonging to the first leg are paid; the ones belonging to the second leg are received.

7.161.2.2 QuantLib::Swap::Swap (const std::vector< Leg > & *legs,* const std::vector< bool > & *payer*)

Multi leg constructor.

7.161.2.3 QuantLib::Swap::Swap (Size *legs*) [protected]

This constructor can be used by derived classes that will build their legs themselves.

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

- C:/quantlib/QuantLib/ql/instruments/**swap.hpp**
- C:/quantlib/QuantLib/ql/instruments/swap.cpp

7.162 QuantLib::Swaption Class Reference

Swaption class

`#include <swaption.hpp>`

Inheritance diagram for QuantLib::Swaption:

Collaboration diagram for QuantLib::Swaption:

Classes

- class **arguments**
  Arguments for swaption calculation
- class **engine**
  base class for swaption engines

Public Member Functions

- **Swaption** (const boost::shared_ptr< **VanillaSwap** > &swap, const boost::shared_ptr< Exercise > &exercise, Settlement::Type delivery=Settlement::Physical)
- Volatility **impliedVolatility** (Real price, const Handle< YieldTermStructure > &discountCurve, Volatility guess, Real accuracy=1.0e-4, Natural maxEvaluations=100, Volatility minVol=1.0e-7, Volatility maxVol=4.0, Real displacement=0.0) const
  implied volatility

Instrument interface

- bool **isExpired** () const
- void **setupArguments** (PricingEngine::arguments *) const

Inspectors

- Settlement::Type **settlementType** () const
- VanillaSwap::Type **type** () const
- const boost::shared_ptr< **VanillaSwap** > & **underlyingSwap** () const

7.162.1 Detailed Description

Swaption class

Test
- the correctness of the returned value is tested by checking that the price of a payer (resp. receiver) swaption decreases (resp. increases) with the strike.
- the correctness of the returned value is tested by checking that the price of a payer (resp. receiver) swaption increases (resp. decreases) with the spread.
- the correctness of the returned value is tested by checking it against that of a swaption on a swap with no spread and a correspondingly adjusted fixed rate.
- the correctness of the returned value is tested by checking it against a known good value.
- the correctness of the returned value of cash settled swaptions is tested by checking the modified annuity against a value calculated without using the **Swaption** (p. 138) class.

Todo add greeks and explicit exercise lag

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

- C:/quantlib/QuantLib/ql/instruments/**swaption.hpp**
- C:/quantlib/QuantLib/ql/instruments/swaption.cpp

7.163 QuantLib::SwingExercise Class Reference

Swing exercise.

`#include <vanillaswingoption.hpp>`

Inheritance diagram for QuantLib::SwingExercise:

Collaboration diagram for QuantLib::SwingExercise:

Public Member Functions

- **SwingExercise** (const std::vector< Date > &dates, const std::vector< Size > &seconds=std::vector< Size >())
- **SwingExercise** (const Date &from, const Date &to, Size stepSizeSecs)
- const std::vector< Size > & **seconds** () const
- std::vector< Time > **exerciseTimes** (const DayCounter &dc, const Date &refDate) const

7.163.1 Detailed Description

Swing exercise.

A Swing option can only be exercised at a set of fixed date times

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

- C:/quantlib/QuantLib/ql/instruments/**vanillaswingoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/**vanillaswingoption.cpp**

7.164 QuantLib::TypePayoff Class Reference

Intermediate class for put/call payoffs.

`#include <payoffs.hpp>`

Inheritance diagram for QuantLib::TypePayoff:

Collaboration diagram for QuantLib::TypePayoff:

Public Member Functions

- Option::Type **optionType** () const

Payoff interface

- std::string **description** () const

Protected Member Functions

- **TypePayoff** (Option::Type type)

Protected Attributes

- Option::Type **type_**

7.164.1 Detailed Description

Intermediate class for put/call payoffs.

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

- C:/quantlib/QuantLib/ql/instruments/**payoffs.hpp**
- C:/quantlib/QuantLib/ql/instruments/payoffs.cpp

7.165 QuantLib::VanillaOption Class Reference

Vanilla option (no discrete dividends, no barriers) on a single asset.

`#include <vanillaoption.hpp>`

Inheritance diagram for QuantLib::VanillaOption:

Collaboration diagram for QuantLib::VanillaOption:

Public Member Functions

- **VanillaOption** (const boost::shared_ptr< **StrikedTypePayoff** > &, const boost::shared_ptr< Exercise > &)
- Volatility **impliedVolatility** (Real price, const boost::shared_ptr< GeneralizedBlackScholesProcess > &process, Real accuracy=1.0e-4, Size maxEvaluations=100, Volatility minVol=1.0e-7, Volatility maxVol=4.0) const

Additional Inherited Members

7.165.1 Detailed Description

Vanilla option (no discrete dividends, no barriers) on a single asset.

7.165.2 Member Function Documentation

7.165.2.1 Volatility QuantLib::VanillaOption::impliedVolatility (Real *price,* const boost::shared_ptr< GeneralizedBlackScholesProcess > & *process,* Real *accuracy =* 1.0e-4, Size *maxEvaluations =* 100, Volatility *minVol =* 1.0e-7, Volatility *maxVol =* 4.0) const

Warning

currently, this method returns the Black-Scholes implied volatility using analytic formulas for European options and a finite-difference method for American and Bermudan options. It will give unconsistent results if the pricing was performed with any other methods (such as jump-diffusion models.)
options with a gamma that changes sign (e.g., binary options) have values that are **not** monotonic in the volatility. In these cases, the calculation can fail and the result (if any) is almost meaningless. Another possible source of failure is to have a target value that is not attainable with any volatility, e.g., a target value lower than the intrinsic value in the case of American options.

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

- C:/quantlib/QuantLib/ql/instruments/**vanillaoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/vanillaoption.cpp

7.166 QuantLib::VanillaStorageOption Class Reference

base option class

`#include <vanillastorageoption.hpp>`

Inheritance diagram for QuantLib::VanillaStorageOption:

Collaboration diagram for QuantLib::VanillaStorageOption:

Classes

- class **arguments**

Public Member Functions

- **VanillaStorageOption** (const boost::shared_ptr< BermudanExercise > &ex, Real capacity, Real load, Real changeRate)
- bool **isExpired** () const
- void **setupArguments** (PricingEngine::arguments *) const

Additional Inherited Members

7.166.1 Detailed Description

base option class

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

- C:/quantlib/QuantLib/ql/instruments/**vanillastorageoption.hpp**

7.167 QuantLib::VanillaSwap Class Reference

Plain-vanilla swap: fix vs floating leg.

`#include <vanillaswap.hpp>`

Inheritance diagram for QuantLib::VanillaSwap:

Collaboration diagram for QuantLib::VanillaSwap:

Classes

- class **arguments**

  Arguments for simple swap calculation
- class **engine**
- class **results**

  Results from simple swap calculation

Public Types

- enum **Type** { **Receiver** = -1, **Payer** = 1 }

Public Member Functions

- **VanillaSwap** (Type type, Real nominal, const Schedule &fixedSchedule, Rate fixedRate, const DayCounter &fixedDayCount, const Schedule &floatSchedule, const boost::shared_ptr< IborIndex > &iborIndex, Spread spread, const DayCounter &floatingDayCount, boost::optional< BusinessDayConvention > paymentConvention=boost::none)
- void **setupArguments** (PricingEngine::arguments *args) const
- void **fetchResults** (const PricingEngine::results *) const

Inspectors

- Type **type** () const
- Real **nominal** () const
- const Schedule & **fixedSchedule** () const
- Rate **fixedRate** () const
- const DayCounter & **fixedDayCount** () const
- const Schedule & **floatingSchedule** () const
- const boost::shared_ptr< IborIndex > & **iborIndex** () const
- Spread **spread** () const
- const DayCounter & **floatingDayCount** () const
- BusinessDayConvention **paymentConvention** () const
- const Leg & **fixedLeg** () const
- const Leg & **floatingLeg** () const

Results

- Real **fixedLegBPS** () const
- Real **fixedLegNPV** () const
- Rate **fairRate** () const
- Real **floatingLegBPS** () const
- Real **floatingLegNPV** () const
- Spread **fairSpread** () const

Additional Inherited Members

7.167.1 Detailed Description

Plain-vanilla swap: fix vs floating leg.

If no payment convention is passed, the convention of the floating-rate schedule is used.

Warning

if `Settings::includeReferenceDateCashFlows()` is set to `true`, payments occurring at the settlement date of the swap might be included in the NPV and therefore affect the fair-rate and fair-spread calculation. This might not be what you want.

Test

- the correctness of the returned value is tested by checking that the price of a swap paying the fair fixed rate is null.
- the correctness of the returned value is tested by checking that the price of a swap receiving the fair floating-rate spread is null.
- the correctness of the returned value is tested by checking that the price of a swap decreases with the paid fixed rate.
- the correctness of the returned value is tested by checking that the price of a swap increases with the received floating-rate spread.
- the correctness of the returned value is tested by checking it against a known good value.

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

- C:/quantlib/QuantLib/ql/instruments/**vanillaswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/vanillaswap.cpp

7.168 QuantLib::VanillaSwingOption Class Reference

base option class

`#include <vanillaswingoption.hpp>`

Inheritance diagram for QuantLib::VanillaSwingOption:

Collaboration diagram for QuantLib::VanillaSwingOption:

Classes

- class **arguments**

Public Member Functions

- **VanillaSwingOption** (const boost::shared_ptr< Payoff > &payoff, const boost::shared_ptr< **SwingExercise** > &ex, Size minExerciseRights, Size maxExerciseRights)
- bool **isExpired** () const
- void **setupArguments** (PricingEngine::arguments *) const

Additional Inherited Members

7.168.1 Detailed Description

base option class

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

- C:/quantlib/QuantLib/ql/instruments/**vanillaswingoption.hpp**
- C:/quantlib/QuantLib/ql/instruments/**vanillaswingoption.cpp**

7.169 QuantLib::VarianceSwap Class Reference

Variance swap.

`#include <varianceswap.hpp>`

Inheritance diagram for QuantLib::VarianceSwap:

Collaboration diagram for QuantLib::VarianceSwap:

Classes

- class **arguments**

  Arguments for forward fair-variance calculation
- class **engine**

  base class for variance-swap engines
- class **results**

  Results from variance-swap calculation

Public Member Functions

- **VarianceSwap** (Position::Type position, Real strike, Real notional, const Date &startDate, const Date &maturityDate)
- void **setupArguments** (PricingEngine::arguments *args) const
- void **fetchResults** (const PricingEngine::results *) const

Instrument interface

- bool **isExpired** () const

Additional interface

- Real **strike** () const
- Position::Type **position** () const
- Date **startDate** () const
- Date **maturityDate** () const
- Real **notional** () const
- Real **variance** () const

Protected Member Functions

- void **setupExpired** () const

Protected Attributes

- Position::Type **position_**
- Real **strike_**
- Real **notional_**
- Date **startDate_**
- Date **maturityDate_**
- Real **variance_**

7.169.1 Detailed Description

Variance swap.

Warning

This class does not manage seasoned variance swaps.

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

- C:/quantlib/QuantLib/ql/instruments/**varianceswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/varianceswap.cpp

7.170 QuantLib::YearOnYearInflationSwap Class Reference

Year-on-year inflation-indexed swap.

`#include <yearonyearinflationswap.hpp>`

Inheritance diagram for QuantLib::YearOnYearInflationSwap:

Collaboration diagram for QuantLib::YearOnYearInflationSwap:

Classes

- class **arguments**

  Arguments for YoY swap calculation
- class **engine**
- class **results**

  Results from YoY swap calculation

Public Types

- enum **Type** { **Receiver** = -1, **Payer** = 1 }

Public Member Functions

- **YearOnYearInflationSwap** (Type type, Real nominal, const Schedule &fixedSchedule, Rate fixedRate, const DayCounter &fixedDayCount, const Schedule &yoySchedule, const boost::shared_ptr< YoYInflationIndex > &yoyIndex, const Period &observationLag, Spread spread, const DayCounter &yoyDayCount, const Calendar &paymentCalendar, BusinessDayConvention paymentConvention=ModifiedFollowing)
- virtual Real **fixedLegNPV** () const
- virtual Rate **fairRate** () const
- virtual Real **yoyLegNPV** () const
- virtual Spread **fairSpread** () const
- virtual Type **type** () const
- virtual Real **nominal** () const
- virtual const Schedule & **fixedSchedule** () const
- virtual Rate **fixedRate** () const
- virtual const DayCounter & **fixedDayCount** () const
- virtual const Schedule & **yoySchedule** () const
- virtual const boost::shared_ptr< YoYInflationIndex > & **yoyInflationIndex** () const
- virtual Period **observationLag** () const
- virtual Spread **spread** () const
- virtual const DayCounter & **yoyDayCount** () const
- virtual Calendar **paymentCalendar** () const
- virtual BusinessDayConvention **paymentConvention** () const
- virtual const Leg & **fixedLeg** () const
- virtual const Leg & **yoyLeg** () const
- void **setupArguments** (PricingEngine::arguments *args) const
- void **fetchResults** (const PricingEngine::results *) const

Additional Inherited Members

7.170.1 Detailed Description

Year-on-year inflation-indexed swap.

Quoted as a fixed rate K. At start:

$$\sum_{i=1}^{M} P_n(0,t_i)NK = \sum_{i=1}^{M} P_n(0,t_i)N\left[\frac{I(t_i)}{I(t_i-1)} - 1\right]$$

where t_M is the maturity time, $P_n(0,t)$ is the nominal discount factor at time t, N is the notional, and $I(t)$ is the inflation index value at time t.

Note

These instruments have now been changed to follow typical **VanillaSwap** (p. 142) type design conventions w.r.t. Schedules etc.

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

- C:/quantlib/QuantLib/ql/instruments/**yearonyearinflationswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/yearonyearinflationswap.cpp

7.171 QuantLib::YoYInflationCap Class Reference

Concrete YoY Inflation cap class.

`#include <inflationcapfloor.hpp>`

Inheritance diagram for QuantLib::YoYInflationCap:

Collaboration diagram for QuantLib::YoYInflationCap:

Public Member Functions

- **YoYInflationCap** (const Leg &yoyLeg, const std::vector< Rate > &exerciseRates)

Additional Inherited Members

7.171.1 Detailed Description

Concrete YoY Inflation cap class.

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

- C:/quantlib/QuantLib/ql/instruments/inflationcapfloor.hpp

7.172 QuantLib::YoYInflationCapFloor Class Reference

Base class for yoy inflation cap-like instruments.

`#include <inflationcapfloor.hpp>`

Inheritance diagram for QuantLib::YoYInflationCapFloor:

Collaboration diagram for QuantLib::YoYInflationCapFloor:

Classes

- class **arguments**

  Arguments for YoY Inflation cap/floor calculation
- class **engine**

  base class for cap/floor engines

Public Types

- enum **Type** { **Cap**, **Floor**, **Collar** }

Public Member Functions

- **YoYInflationCapFloor** (Type type, const Leg &yoyLeg, const std::vector< Rate > &capRates, const std::vector< Rate > &floorRates)
- **YoYInflationCapFloor** (Type type, const Leg &yoyLeg, const std::vector< Rate > &strikes)
- virtual Rate **atmRate** (const YieldTermStructure &discountCurve) const
- virtual Volatility **impliedVolatility** (Real price, const Handle< YoYInflationTermStructure > &yoyCurve, Volatility guess, Real accuracy=1.0e-4, Natural maxEvaluations=100, Volatility minVol=1.0e-7, Volatility maxVol=4.0) const

  implied term volatility

Instrument interface

- bool **isExpired** () const
- void **setupArguments** (PricingEngine::arguments *) const

Inspectors

- Type **type** () const
- const std::vector< Rate > & **capRates** () const
- const std::vector< Rate > & **floorRates** () const
- const Leg & **yoyLeg** () const
- Date **startDate** () const
- Date **maturityDate** () const
- boost::shared_ptr< YoYInflationCoupon > **lastYoYInflationCoupon** () const
- boost::shared_ptr< **YoYInflationCapFloor** > **optionlet** (const Size n) const

  Returns the n-th optionlet as a cap/floor with only one cash flow.

7.172.1 Detailed Description

Base class for yoy inflation cap-like instruments.

Note that the standard YoY inflation cap/floor defined here is different from nominal, because in nominal world standard cap/floors do not have the first optionlet. This is because they set in advance so there is no point. However, yoy inflation generally sets (effectively) in arrears, (actually in arrears vs lag of a few months) thus the first optionlet is relevant. Hence we can do a parity test without a special definition of the YoY cap/floor instrument.

Test
- the relationship between the values of caps, floors and the resulting collars is checked.
- the put-call parity between the values of caps, floors and swaps is checked.
- the correctness of the returned value is tested by checking it against a known good value.

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

- C:/quantlib/QuantLib/ql/instruments/inflationcapfloor.hpp
- C:/quantlib/QuantLib/ql/instruments/inflationcapfloor.cpp

7.173 QuantLib::YoYInflationCollar Class Reference

Concrete YoY Inflation collar class.

`#include <inflationcapfloor.hpp>`

Inheritance diagram for QuantLib::YoYInflationCollar:

Collaboration diagram for QuantLib::YoYInflationCollar:

Public Member Functions

- **YoYInflationCollar** (const Leg &yoyLeg, const std::vector< Rate > &capRates, const std::vector< Rate > &floorRates)

Additional Inherited Members

7.173.1 Detailed Description

Concrete YoY Inflation collar class.

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

- C:/quantlib/QuantLib/ql/instruments/inflationcapfloor.hpp

7.174 QuantLib::YoYInflationFloor Class Reference

Concrete YoY Inflation floor class.

`#include <inflationcapfloor.hpp>`

Inheritance diagram for QuantLib::YoYInflationFloor:

Collaboration diagram for QuantLib::YoYInflationFloor:

Public Member Functions

- **YoYInflationFloor** (const Leg &yoyLeg, const std::vector< Rate > &exerciseRates)

Additional Inherited Members

7.174.1 Detailed Description

Concrete YoY Inflation floor class.

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

- C:/quantlib/QuantLib/ql/instruments/inflationcapfloor.hpp

7.175 QuantLib::ZeroCouponBond Class Reference

zero-coupon bond

`#include <zerocouponbond.hpp>`

Inheritance diagram for QuantLib::ZeroCouponBond:

Collaboration diagram for QuantLib::ZeroCouponBond:

Public Member Functions

- **ZeroCouponBond** (Natural settlementDays, const Calendar &calendar, Real faceAmount, const Date &maturityDate, BusinessDayConvention paymentConvention=Following, Real **redemption**=100.0, const Date &issueDate=Date())

Additional Inherited Members

7.175.1 Detailed Description

zero-coupon bond

Test calculations are tested by checking results against cached values.

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

- C:/quantlib/QuantLib/ql/instruments/bonds/**zerocouponbond.hpp**
- C:/quantlib/QuantLib/ql/instruments/bonds/zerocouponbond.cpp

7.176 QuantLib::ZeroCouponInflationSwap Class Reference

Zero-coupon inflation-indexed swap.

`#include <zerocouponinflationswap.hpp>`

Inheritance diagram for QuantLib::ZeroCouponInflationSwap:

Collaboration diagram for QuantLib::ZeroCouponInflationSwap:

Classes

- class **arguments**
- class **engine**

Public Types

- enum **Type** { **Receiver** = -1, **Payer** = 1 }

Public Member Functions

- **ZeroCouponInflationSwap** (Type **type**, Real nominal, const Date &startDate, const Date &maturity, const Calendar &fixCalendar, BusinessDayConvention fixConvention, const DayCounter &dayCounter, Rate **fixedRate**, const boost::shared_ptr< ZeroInflationIndex > &infIndex, const Period &observation↩ Lag, bool adjustInfObsDates=false, Calendar infCalendar=Calendar(), BusinessDayConvention inf↩ Convention=BusinessDayConvention())

Inspectors

- Type **type** () const
  "payer" or "receiver" refer to the inflation-indexed leg
- Real **nominal** () const
- Date **startDate** () const
- Date **maturityDate** () const
- Calendar **fixedCalendar** () const
- BusinessDayConvention **fixedConvention** () const
- DayCounter **dayCounter** () const
- Rate **fixedRate** () const
  K *in the above formula.*
- boost::shared_ptr< ZeroInflationIndex > **inflationIndex** () const
- Period **observationLag** () const
- bool **adjustObservationDates** () const
- Calendar **inflationCalendar** () const
- BusinessDayConvention **inflationConvention** () const
- const Leg & **fixedLeg** () const
  just one cashflow (that is not a coupon) in each leg
- const Leg & **inflationLeg** () const
  just one cashflow (that is not a coupon) in each leg

Instrument interface

- void **setupArguments** (PricingEngine::arguments *) const
- void **fetchResults** (const PricingEngine::results *r) const

Results

- Real **fixedLegNPV** () const
- Real **inflationLegNPV** () const
- Real **fairRate** () const

Protected Attributes

- Type **type_**
- Real **nominal_**
- Date **startDate_**
- Date **maturityDate_**
- Calendar **fixCalendar_**
- BusinessDayConvention **fixConvention_**
- Rate **fixedRate_**
- boost::shared_ptr< ZeroInflationIndex > **inflndex_**
- Period **observationLag_**
- bool **adjustInfObsDates_**
- Calendar **infCalendar_**
- BusinessDayConvention **infConvention_**
- DayCounter **dayCounter_**
- Date **baseDate_**
- Date **obsDate_**

Additional Inherited Members

7.176.1 Detailed Description

Zero-coupon inflation-indexed swap.

Quoted as a fixed rate K. At start:

$$P_n(0,T)N[(1+K)^T - 1] = P_n(0,T)N\left[\frac{I(T)}{I(0)} - 1\right]$$

where T is the maturity time, $P_n(0,t)$ is the nominal discount factor at time t, N is the notional, and $I(t)$ is the inflation index value at time t.

This inherits from swap and has two very simple legs: a fixed leg, from the quote (K); and an indexed leg. At maturity the two single cashflows are swapped. These are the notional versus the inflation-indexed notional Because the coupons are zero there are no accruals (and no coupons).

Inflation is generally available on every day, including holidays and weekends. Hence there is a variable to state whether the observe/fix dates for inflation are adjusted or not. The default is not to adjust.

A zero inflation swap is a simple enough instrument that the standard discounting pricing engine that works for a vanilla swap also works.

Note

we do not need Schedules on the legs because they use one or two dates only per leg.

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

- C:/quantlib/QuantLib/ql/instruments/**zerocouponinflationswap.hpp**
- C:/quantlib/QuantLib/ql/instruments/zerocouponinflationswap.cpp

Chapter 8

File Documentation

8.1 C:/quantlib/QuantLib/ql/instruments/asianoption.hpp File Reference

Asian option on a single asset.

```
#include <ql/instruments/oneassetoption.hpp>
#include <ql/instruments/payoffs.hpp>
#include <ql/instruments/averagetype.hpp>
#include <ql/time/date.hpp>
#include <vector>
```

Include dependency graph for asianoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::ContinuousAveragingAsianOption**

  Continuous-averaging Asian option.
- class **QuantLib::DiscreteAveragingAsianOption**

  Discrete-averaging Asian option.
- class **QuantLib::DiscreteAveragingAsianOption::arguments**

  Extra arguments for single-asset discrete-average Asian option.
- class **QuantLib::ContinuousAveragingAsianOption::arguments**

  Extra arguments for single-asset continuous-average Asian option.
- class **QuantLib::DiscreteAveragingAsianOption::engine**

  Discrete-averaging Asian engine base class.
- class **QuantLib::ContinuousAveragingAsianOption::engine**

  Continuous-averaging Asian engine base class.

8.1.1 Detailed Description

Asian option on a single asset.

8.2 C:/quantlib/QuantLib/ql/instruments/assetswap.hpp File Reference

Bullet bond vs Libor swap.

```
#include <ql/instruments/swap.hpp>
#include <ql/instruments/bond.hpp>
#include <ql/time/schedule.hpp>
#include <ql/time/daycounter.hpp>
```

Include dependency graph for assetswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::AssetSwap**

  Bullet bond vs Libor swap.
- class **QuantLib::AssetSwap::arguments**

  Arguments for asset swap calculation
- class **QuantLib::AssetSwap::results**

  Results from simple swap calculation

8.2.1 Detailed Description

Bullet bond vs Libor swap.

8.3 C:/quantlib/QuantLib/ql/instruments/averagetype.hpp File Reference

Averaging algorithm enumeration.

```
#include <ql/qldefines.hpp>
#include <iosfwd>
```

Include dependency graph for averagetype.hpp: This graph shows which files directly or indirectly include this file:

Classes

- struct **QuantLib::Average**

  Placeholder for enumerated averaging types.

Functions

- std::ostream & **QuantLib::operator**<< (std::ostream &out, Average::Type type)

8.3.1 Detailed Description

Averaging algorithm enumeration.

8.4 C:/quantlib/QuantLib/ql/instruments/barrieroption.hpp File Reference

Barrier option on a single asset.

```
#include <ql/instruments/oneassetoption.hpp>
#include <ql/instruments/barriertype.hpp>
#include <ql/instruments/payoffs.hpp>
```

Include dependency graph for barrieroption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::BarrierOption**

  Barrier option on a single asset.
- class **QuantLib::BarrierOption::arguments**

  Arguments for barrier option calculation
- class **QuantLib::BarrierOption::engine**

  Barrier-option engine base class

8.4.1 Detailed Description

Barrier option on a single asset.

8.5 C:/quantlib/QuantLib/ql/instruments/barriertype.hpp File Reference

Barrier type.

```
#include <ql/qldefines.hpp>
#include <iosfwd>
```

Include dependency graph for barriertype.hpp: This graph shows which files directly or indirectly include this file:

Classes

- struct **QuantLib::Barrier**

  Placeholder for enumerated barrier types.

Functions

- std::ostream & **QuantLib::operator**<< (std::ostream &out, Barrier::Type type)

8.5.1 Detailed Description

Barrier type.

8.6 C:/quantlib/QuantLib/ql/instruments/basketoption.hpp File Reference

Basket option on a number of assets.

```
#include <ql/instruments/payoffs.hpp>
#include <ql/instruments/multiassetoption.hpp>
#include <ql/math/array.hpp>
```

Include dependency graph for basketoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::BasketPayoff**
- class **QuantLib::MinBasketPayoff**
- class **QuantLib::MaxBasketPayoff**
- class **QuantLib::AverageBasketPayoff**
- class **QuantLib::SpreadBasketPayoff**
- class **QuantLib::BasketOption**

  Basket option on a number of assets.
- class **QuantLib::BasketOption::engine**

  Basket-option engine base class

8.6.1 Detailed Description

Basket option on a number of assets.

8.7 C:/quantlib/QuantLib/ql/instruments/bmaswap.hpp File Reference

swap paying Libor against BMA coupons

```
#include <ql/instruments/swap.hpp>
#include <ql/indexes/iborindex.hpp>
#include <ql/indexes/bmaindex.hpp>
```

Include dependency graph for bmaswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::BMASwap**

  swap paying Libor against BMA coupons

8.7.1 Detailed Description

swap paying Libor against BMA coupons

8.8 C:/quantlib/QuantLib/ql/instruments/bond.hpp File Reference

concrete bond class

```
#include <ql/instrument.hpp>
#include <ql/time/calendar.hpp>
#include <ql/cashflow.hpp>
#include <ql/compounding.hpp>
#include <vector>
```

Include dependency graph for bond.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::Bond**

  Base bond class.
- class **QuantLib::Bond::arguments**
- class **QuantLib::Bond::results**
- class **QuantLib::Bond::engine**

8.8.1 Detailed Description

concrete bond class

8.9 C:/quantlib/QuantLib/ql/instruments/bonds/btp.hpp File Reference

Italian BTP (Buoni Poliennali del Tesoro) fixed rate bond.

```
#include <ql/instruments/bonds/fixedratebond.hpp>
#include <ql/instruments/bonds/floatingratebond.hpp>
#include <ql/indexes/ibor/euribor.hpp>
#include <ql/instruments/vanillaswap.hpp>
#include <numeric>
```

Include dependency graph for btp.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::CCTEU**
- class **QuantLib::BTP**

  *Italian **BTP** (p. 51) (Buono Poliennali del Tesoro) fixed rate bond.*
- class **QuantLib::RendistatoBasket**
- class **QuantLib::RendistatoCalculator**
- class **QuantLib::RendistatoEquivalentSwapLengthQuote**

  ***RendistatoCalculator** (p. 122) equivalent swap lenth Quote adapter.*
- class **QuantLib::RendistatoEquivalentSwapSpreadQuote**

  ***RendistatoCalculator** (p. 122) equivalent swap spread Quote adapter.*

8.9.1 Detailed Description

Italian BTP (Buoni Poliennali del Tesoro) fixed rate bond.

8.10 C:/quantlib/QuantLib/ql/instruments/bonds/cmsratebond.hpp File Reference

CMS-rate bond.

```
#include <ql/instruments/bond.hpp>
```

Include dependency graph for cmsratebond.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::CmsRateBond**

  CMS-rate bond.

8.10.1 Detailed Description

CMS-rate bond.

8.11 C:/quantlib/QuantLib/ql/instruments/bonds/cpibond.hpp File Reference

zero-inflation-indexed-ratio-with-base bond

```
#include <ql/instruments/bond.hpp>
#include <ql/time/dategenerationrule.hpp>
#include <ql/time/daycounter.hpp>
#include <ql/interestrate.hpp>
#include <ql/cashflows/cpicoupon.hpp>
```

Include dependency graph for cpibond.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::CPIBond**

8.11.1 Detailed Description

zero-inflation-indexed-ratio-with-base bond

8.12 C:/quantlib/QuantLib/ql/instruments/bonds/fixedratebond.hpp File Reference

fixed-rate bond

```
#include <ql/instruments/bond.hpp>
#include <ql/time/dategenerationrule.hpp>
#include <ql/time/daycounter.hpp>
#include <ql/interestrate.hpp>
```

Include dependency graph for fixedratebond.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::FixedRateBond**
  fixed-rate bond

8.12.1 Detailed Description

fixed-rate bond

8.13 C:/quantlib/QuantLib/ql/instruments/bonds/floatingratebond.hpp File Reference

floating-rate bond

```
#include <ql/instruments/bond.hpp>
#include <ql/time/dategenerationrule.hpp>
```

Include dependency graph for floatingratebond.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::FloatingRateBond**
  floating-rate bond (possibly capped and/or floored)

8.13.1 Detailed Description

floating-rate bond

8.14 C:/quantlib/QuantLib/ql/instruments/bonds/zerocouponbond.hpp File Reference

zero-coupon bond

```
#include <ql/instruments/bond.hpp>
```

Include dependency graph for zerocouponbond.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::ZeroCouponBond**

  zero-coupon bond

8.14.1 Detailed Description

zero-coupon bond

8.15 C:/quantlib/QuantLib/ql/instruments/callabilityschedule.hpp File Reference

Schedule of put/call dates.

```
#include <ql/event.hpp>
#include <ql/patterns/visitor.hpp>
#include <ql/utilities/null.hpp>
#include <boost/shared_ptr.hpp>
#include <boost/optional.hpp>
#include <vector>
```

Include dependency graph for callabilityschedule.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::Callability**

  instrument callability
- class **QuantLib::Callability::Price**

  amount to be paid upon callability

Typedefs

- typedef std::vector< boost::shared_ptr< Callability > > **QuantLib::CallabilitySchedule**

8.15.1 Detailed Description

Schedule of put/call dates.

8.16 C:/quantlib/QuantLib/ql/instruments/capfloor.hpp File Reference

cap and floor class

```
#include <ql/instrument.hpp>
#include <ql/cashflows/iborcoupon.hpp>
#include <ql/handle.hpp>
```

Include dependency graph for capfloor.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::CapFloor**

  Base class for cap-like instruments.
- class **QuantLib::Cap**

  Concrete cap class.
- class **QuantLib::Floor**

  Concrete floor class.
- class **QuantLib::Collar**

  Concrete collar class.
- class **QuantLib::CapFloor::arguments**

  Arguments for cap/floor calculation
- class **QuantLib::CapFloor::engine**

  base class for cap/floor engines

Functions

- std::ostream & **QuantLib::operator**<< (std::ostream &out, CapFloor::Type t)

8.16.1 Detailed Description

cap and floor class

inflation cap and floor class, just year-on-year variety for now

8.17 C:/quantlib/QuantLib/ql/instruments/claim.hpp File Reference

Classes for default-event claims.

`#include <ql/instruments/bond.hpp>`
Include dependency graph for claim.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::Claim**

  Claim *(*p. *56) associated to a default event.*
- class **QuantLib::FaceValueClaim**

  Claim *(*p. *56) on a notional.*
- class **QuantLib::FaceValueAccrualClaim**

  Claim *(*p. *56) on the notional of a reference security, including accrual.*

8.17.1 Detailed Description

Classes for default-event claims.

8.18 C:/quantlib/QuantLib/ql/instruments/cliquetoption.hpp File Reference

Cliquet option.

```
#include <ql/instruments/oneassetoption.hpp>
#include <ql/instruments/payoffs.hpp>
#include <ql/time/date.hpp>
#include <vector>
```

Include dependency graph for cliquetoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::CliquetOption**

  cliquet (Ratchet) option
- class **QuantLib::CliquetOption::arguments**

  Arguments for cliquet option calculation
- class **QuantLib::CliquetOption::engine**

  Cliquet engine base class.

8.18.1 Detailed Description

Cliquet option.

8.19 C:/quantlib/QuantLib/ql/instruments/compositeinstrument.hpp File Reference

Composite instrument class.

```
#include <ql/instrument.hpp>
#include <list>
#include <utility>
```

Include dependency graph for compositeinstrument.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::CompositeInstrument**

  Composite instrument

8.19.1 Detailed Description

Composite instrument class.

8.20 C:/quantlib/QuantLib/ql/instruments/cpicapfloor.hpp File Reference

zero-inflation-indexed-ratio-with-base option

```
#include <ql/instrument.hpp>
#include <ql/option.hpp>
#include <ql/time/calendar.hpp>
#include <ql/time/daycounter.hpp>
#include <ql/indexes/inflationindex.hpp>
#include <ql/cashflows/cpicoupon.hpp>
```

Include dependency graph for cpicapfloor.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::CPICapFloor**

  CPI cap or floor.
- class **QuantLib::CPICapFloor::arguments**
- class **QuantLib::CPICapFloor::results**
- class **QuantLib::CPICapFloor::engine**

8.20.1 Detailed Description

zero-inflation-indexed-ratio-with-base option

8.21 C:/quantlib/QuantLib/ql/instruments/cpiswap.hpp File Reference

zero-inflation-indexed-ratio-with-base swap

```
#include <ql/instruments/swap.hpp>
#include <ql/time/calendar.hpp>
#include <ql/time/daycounter.hpp>
#include <ql/time/schedule.hpp>
#include <ql/indexes/iborindex.hpp>
#include <ql/cashflows/cpicoupon.hpp>
```

Include dependency graph for cpiswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::CPISwap**

  zero-inflation-indexed swap,
- class **QuantLib::CPISwap::arguments**

  Arguments for swap calculation
- class **QuantLib::CPISwap::results**

  Results from swap calculation
- class **QuantLib::CPISwap::engine**

Functions

- std::ostream & **QuantLib::operator**<< (std::ostream &out, CPISwap::Type t)

8.21.1 Detailed Description

zero-inflation-indexed-ratio-with-base swap

8.22 C:/quantlib/QuantLib/ql/instruments/creditdefaultswap.hpp File Reference

Credit default swap.

```
#include <ql/instrument.hpp>
#include <ql/cashflow.hpp>
#include <ql/default.hpp>
#include <ql/termstructures/defaulttermstructure.hpp>
#include <ql/time/schedule.hpp>
```

Include dependency graph for creditdefaultswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::CreditDefaultSwap**

  Credit default swap.
- class **QuantLib::CreditDefaultSwap::arguments**
- class **QuantLib::CreditDefaultSwap::results**
- class **QuantLib::CreditDefaultSwap::engine**

8.22.1 Detailed Description

Credit default swap.

8.23 C:/quantlib/QuantLib/ql/instruments/dividendbarrieroption.hpp File Reference

Barrier option on a single asset with discrete dividends.

```
#include <ql/instruments/barrieroption.hpp>
#include <ql/instruments/dividendschedule.hpp>
#include <ql/instruments/payoffs.hpp>
```

Include dependency graph for dividendbarrieroption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::DividendBarrierOption**
  Single-asset barrier option with discrete dividends.
- class **QuantLib::DividendBarrierOption::arguments**
  Arguments for dividend barrier option calculation
- class **QuantLib::DividendBarrierOption::engine**
  Dividend-barrier-option engine base class

8.23.1 Detailed Description

Barrier option on a single asset with discrete dividends.

8.24 C:/quantlib/QuantLib/ql/instruments/dividendschedule.hpp File Reference

Schedule of dividend dates.

```
#include <ql/cashflows/dividend.hpp>
#include <vector>
```

Include dependency graph for dividendschedule.hpp: This graph shows which files directly or indirectly include this file:

Typedefs

- typedef std::vector< boost::shared_ptr< Dividend > > **QuantLib::DividendSchedule**

8.24.1 Detailed Description

Schedule of dividend dates.

8.25 C:/quantlib/QuantLib/ql/instruments/dividendvanillaoption.hpp File Reference

Vanilla option on a single asset with discrete dividends.

```
#include <ql/instruments/oneassetoption.hpp>
#include <ql/instruments/dividendschedule.hpp>
#include <ql/instruments/payoffs.hpp>
```

Include dependency graph for dividendvanillaoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::DividendVanillaOption**
  Single-asset vanilla option (no barriers) with discrete dividends.
- class **QuantLib::DividendVanillaOption::arguments**
  Arguments for dividend vanilla option calculation
- class **QuantLib::DividendVanillaOption::engine**
  Dividend-vanilla-option engine base class

8.25.1 Detailed Description

Vanilla option on a single asset with discrete dividends.

8.26 C:/quantlib/QuantLib/ql/instruments/europeanoption.hpp File Reference

European option on a single asset.

```
#include <ql/instruments/vanillaoption.hpp>
```

Include dependency graph for europeanoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::EuropeanOption**

  European option on a single asset.

8.26.1 Detailed Description

European option on a single asset.

8.27 C:/quantlib/QuantLib/ql/instruments/fixedratebondforward.hpp File Reference

forward contract on a fixed-rate bond

```
#include <ql/instruments/forward.hpp>
#include <ql/instruments/bonds/fixedratebond.hpp>
```

Include dependency graph for fixedratebondforward.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::FixedRateBondForward**

  Forward contract on a fixed-rate bond

8.27.1 Detailed Description

forward contract on a fixed-rate bond

8.28 C:/quantlib/QuantLib/ql/instruments/floatfloatswap.hpp File Reference

swap exchanging capped floored Libor or CMS coupons with quite general specification. If no payment convention is given, the respective leg schedule convention is used. The interest rate indices should be linked to valid forwarding and in case of swap indices discounting curves

```
#include <ql/instruments/swap.hpp>
#include <ql/instruments/vanillaswap.hpp>
#include <ql/time/daycounter.hpp>
#include <ql/time/schedule.hpp>
#include <boost/optional.hpp>
```

Include dependency graph for floatfloatswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::FloatFloatSwap**
  float float swap
- class **QuantLib::FloatFloatSwap::arguments**
  Arguments for float float swap calculation
- class **QuantLib::FloatFloatSwap::results**
  Results from float float swap calculation
- class **QuantLib::FloatFloatSwap::engine**

8.28.1 Detailed Description

swap exchanging capped floored Libor or CMS coupons with quite general specification. If no payment convention is given, the respective leg schedule convention is used. The interest rate indices should be linked to valid forwarding and in case of swap indices discounting curves

8.29 C:/quantlib/QuantLib/ql/instruments/floatfloatswaption.hpp File Reference

floatfloatswaption class

```
#include <ql/option.hpp>
#include <ql/instruments/floatfloatswap.hpp>
#include <ql/pricingengines/swaption/basketgeneratingengine.hpp>
#include <ql/termstructures/yieldtermstructure.hpp>
#include <ql/termstructures/volatility/swaption/swaptionvolstructure.hpp>
#include <ql/models/calibrationhelper.hpp>
#include <ql/utilities/disposable.hpp>
```

Include dependency graph for floatfloatswaption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::FloatFloatSwaption**
  floatfloat swaption class
- class **QuantLib::FloatFloatSwaption::arguments**
  Arguments for cms swaption calculation
- class **QuantLib::FloatFloatSwaption::engine**
  base class for cms swaption engines

8.29.1 Detailed Description

floatfloatswaption class

8.30 C:/quantlib/QuantLib/ql/instruments/forward.hpp File Reference

Base forward class.

```
#include <ql/instrument.hpp>
#include <ql/position.hpp>
#include <ql/time/calendar.hpp>
#include <ql/time/daycounter.hpp>
#include <ql/interestrate.hpp>
#include <ql/types.hpp>
#include <ql/handle.hpp>
#include <ql/payoff.hpp>
#include <ql/termstructures/yieldtermstructure.hpp>
```

Include dependency graph for forward.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::Forward**

  Abstract base forward class.
- class **QuantLib::ForwardTypePayoff**

  Class for forward type payoffs.

8.30.1 Detailed Description

Base forward class.

8.31 C:/quantlib/QuantLib/ql/instruments/forwardrateagreement.hpp File Reference

forward rate agreement

```
#include <ql/instruments/forward.hpp>
```

Include dependency graph for forwardrateagreement.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::ForwardRateAgreement**

8.31.1 Detailed Description

forward rate agreement

8.32 C:/quantlib/QuantLib/ql/instruments/forwardvanillaoption.hpp File Reference

Forward version of a vanilla option.

```
#include <ql/instruments/oneassetoption.hpp>
#include <ql/instruments/payoffs.hpp>
#include <ql/exercise.hpp>
#include <ql/settings.hpp>
```

Include dependency graph for forwardvanillaoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::ForwardOptionArguments**< **ArgumentsType** >

  Arguments for forward (strike-resetting) option calculation
- class **QuantLib::ForwardVanillaOption**

  Forward version of a vanilla option

8.32.1 Detailed Description

Forward version of a vanilla option.

8.33 C:/quantlib/QuantLib/ql/instruments/futures.hpp File Reference

Futures.

```
#include <ql/qldefines.hpp>
#include <iosfwd>
```

Include dependency graph for futures.hpp: This graph shows which files directly or indirectly include this file:

Classes

- struct **QuantLib::Futures**

8.33.1 Detailed Description

Futures.

8.34 C:/quantlib/QuantLib/ql/instruments/impliedvolatility.hpp File Reference

Utilities for implied-volatility calculation.

```
#include <ql/instrument.hpp>
#include <ql/quotes/simplequote.hpp>
#include <ql/processes/blackscholesprocess.hpp>
```

Include dependency graph for impliedvolatility.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::detail::ImpliedVolatilityHelper**

  helper class for one-asset implied-volatility calculation

8.34.1 Detailed Description

Utilities for implied-volatility calculation.

8.35 C:/quantlib/QuantLib/ql/instruments/lookbackoption.hpp File Reference

Lookback option on a single asset.

```
#include <ql/instruments/oneassetoption.hpp>
#include <ql/instruments/payoffs.hpp>
#include <ql/exercise.hpp>
```

Include dependency graph for lookbackoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::ContinuousFloatingLookbackOption**

  Continuous-floating lookback option.
- class **QuantLib::ContinuousFixedLookbackOption**

  Continuous-fixed lookback option.
- class **QuantLib::ContinuousPartialFloatingLookbackOption**

  Continuous-partial-floating lookback option.
- class **QuantLib::ContinuousPartialFixedLookbackOption**

  Continuous-partial-fixed lookback option.
- class **QuantLib::ContinuousFloatingLookbackOption::arguments**

  Arguments for continuous floating lookback option calculation
- class **QuantLib::ContinuousFixedLookbackOption::arguments**

  Arguments for continuous fixed lookback option calculation
- class **QuantLib::ContinuousPartialFloatingLookbackOption::arguments**

  Arguments for continuous partial floating lookback option calculation
- class **QuantLib::ContinuousPartialFixedLookbackOption::arguments**

  Arguments for continuous partial fixed lookback option calculation
- class **QuantLib::ContinuousFloatingLookbackOption::engine**

  Continuous floating lookback engine base class
- class **QuantLib::ContinuousFixedLookbackOption::engine**

  Continuous fixed lookback engine base class
- class **QuantLib::ContinuousPartialFloatingLookbackOption::engine**

  Continuous partial floating lookback engine base class
- class **QuantLib::ContinuousPartialFixedLookbackOption::engine**

  Continuous partial fixed lookback engine base class

8.35.1 Detailed Description

Lookback option on a single asset.

8.36 C:/quantlib/QuantLib/ql/instruments/makecapfloor.hpp File Reference

Helper class to instantiate standard market cap/floor.

```
#include <ql/instruments/capfloor.hpp>
#include <ql/instruments/makevanillaswap.hpp>
```

Include dependency graph for makecapfloor.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::MakeCapFloor**

  helper class

8.36.1 Detailed Description

Helper class to instantiate standard market cap/floor.

Helper class to instantiate standard yoy inflation cap/floor.

8.37 C:/quantlib/QuantLib/ql/instruments/makecms.hpp File Reference

Helper class to instantiate standard market CMS.

```
#include <ql/cashflows/cmscoupon.hpp>
#include <ql/cashflows/couponpricer.hpp>
#include <ql/pricingengine.hpp>
```

Include dependency graph for makecms.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::MakeCms**

  helper class for instantiating CMS

8.37.1 Detailed Description

Helper class to instantiate standard market CMS.

8.38 C:/quantlib/QuantLib/ql/instruments/makeois.hpp File Reference

Helper class to instantiate overnight indexed swaps.

```
#include <ql/instruments/overnightindexedswap.hpp>
#include <ql/time/dategenerationrule.hpp>
#include <ql/termstructures/yieldtermstructure.hpp>
```

Include dependency graph for makeois.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::MakeOIS**

  helper class

8.38.1 Detailed Description

Helper class to instantiate overnight indexed swaps.

8.39 C:/quantlib/QuantLib/ql/instruments/makeswaption.hpp File Reference

Helper class to instantiate standard market swaption.

```
#include <ql/time/businessdayconvention.hpp>
#include <ql/instruments/swaption.hpp>
```

Include dependency graph for makeswaption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::MakeSwaption**

  helper class

8.39.1 Detailed Description

Helper class to instantiate standard market swaption.

8.40 C:/quantlib/QuantLib/ql/instruments/makevanillaswap.hpp File Reference

Helper class to instantiate standard market swaps.

```
#include <ql/instruments/vanillaswap.hpp>
#include <ql/time/dategenerationrule.hpp>
#include <ql/termstructures/yieldtermstructure.hpp>
```

Include dependency graph for makevanillaswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::MakeVanillaSwap**

  helper class

8.40.1 Detailed Description

Helper class to instantiate standard market swaps.

8.41 C:/quantlib/QuantLib/ql/instruments/multiassetoption.hpp File Reference

Option on multiple assets.

```
#include <ql/option.hpp>
```

Include dependency graph for multiassetoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::MultiAssetOption**

  Base class for options on multiple assets.
- class **QuantLib::MultiAssetOption::results**

  Results from multi-asset option calculation
- class **QuantLib::MultiAssetOption::engine**

8.41.1 Detailed Description

Option on multiple assets.

8.42 C:/quantlib/QuantLib/ql/instruments/nonstandardswap.hpp File Reference

vanilla swap but possibly with period dependent nominal and strike

```
#include <ql/instruments/swap.hpp>
#include <ql/instruments/vanillaswap.hpp>
#include <ql/time/daycounter.hpp>
#include <ql/time/schedule.hpp>
#include <boost/optional.hpp>
```

Include dependency graph for nonstandardswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::NonstandardSwap**
  nonstandard swap
- class **QuantLib::NonstandardSwap::arguments**
  Arguments for nonstandard swap calculation
- class **QuantLib::NonstandardSwap::results**
  Results from nonstandard swap calculation
- class **QuantLib::NonstandardSwap::engine**

8.42.1 Detailed Description

vanilla swap but possibly with period dependent nominal and strike

8.43 C:/quantlib/QuantLib/ql/instruments/nonstandardswaption.hpp File Reference

nonstandard swap option class

```
#include <ql/option.hpp>
#include <ql/instruments/swaption.hpp>
#include <ql/instruments/nonstandardswap.hpp>
#include <ql/pricingengines/swaption/basketgeneratingengine.hpp>
#include <ql/termstructures/yieldtermstructure.hpp>
#include <ql/termstructures/volatility/swaption/swaptionvolstructure.hpp>
#include <ql/models/calibrationhelper.hpp>
#include <ql/utilities/disposable.hpp>
```

Include dependency graph for nonstandardswaption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::NonstandardSwaption**
  nonstandard swaption class
- class **QuantLib::NonstandardSwaption::arguments**
  Arguments for nonstandard swaption calculation
- class **QuantLib::NonstandardSwaption::engine**
  base class for nonstandard swaption engines

8.43.1 Detailed Description

nonstandard swap option class

8.44 C:/quantlib/QuantLib/ql/instruments/oneassetoption.hpp File Reference

Option on a single asset.

```
#include <ql/option.hpp>
```

Include dependency graph for oneassetoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::OneAssetOption**

  Base class for options on a single asset.
- class **QuantLib::OneAssetOption::results**

  Results from single-asset option calculation
- class **QuantLib::OneAssetOption::engine**

8.44.1 Detailed Description

Option on a single asset.

8.45 C:/quantlib/QuantLib/ql/instruments/overnightindexedswap.hpp File Reference

Overnight index swap paying compounded overnight vs. fixed.

```
#include <ql/instruments/swap.hpp>
#include <ql/time/daycounter.hpp>
```

Include dependency graph for overnightindexedswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::OvernightIndexedSwap**

  Overnight indexed swap: fix vs compounded overnight rate.

8.45.1 Detailed Description

Overnight index swap paying compounded overnight vs. fixed.

8.46 C:/quantlib/QuantLib/ql/instruments/payoffs.hpp File Reference

Payoffs for various options.

```
#include <ql/option.hpp>
#include <ql/payoff.hpp>
```

Include dependency graph for payoffs.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::NullPayoff**
  Dummy payoff class.
- class **QuantLib::TypePayoff**
  Intermediate class for put/call payoffs.
- class **QuantLib::FloatingTypePayoff**
  Payoff based on a floating strike
- class **QuantLib::StrikedTypePayoff**
  Intermediate class for payoffs based on a fixed strike.
- class **QuantLib::PlainVanillaPayoff**
  Plain-vanilla payoff.
- class **QuantLib::PercentageStrikePayoff**
  Payoff with strike expressed as percentage
- class **QuantLib::AssetOrNothingPayoff**
  Binary asset-or-nothing payoff.
- class **QuantLib::CashOrNothingPayoff**
  Binary cash-or-nothing payoff.
- class **QuantLib::GapPayoff**
  Binary gap payoff.
- class **QuantLib::SuperFundPayoff**
  Binary supershare and superfund payoffs.
- class **QuantLib::SuperSharePayoff**
  Binary supershare payoff.

8.46.1 Detailed Description

Payoffs for various options.

8.47 C:/quantlib/QuantLib/ql/instruments/quantobarrieroption.hpp File Reference

Quanto version of a barrier option.

```
#include <ql/instruments/quantovanillaoption.hpp>
#include <ql/instruments/barrieroption.hpp>
```

Include dependency graph for quantobarrieroption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::QuantoBarrierOption**
  Quanto version of a barrier option.

8.47.1 Detailed Description

Quanto version of a barrier option.

8.48 C:/quantlib/QuantLib/ql/instruments/quantoforwardvanillaoption.hpp File Reference

Quanto version of a forward vanilla option.

```
#include <ql/instruments/quantovanillaoption.hpp>
#include <ql/instruments/forwardvanillaoption.hpp>
```

Include dependency graph for quantoforwardvanillaoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::QuantoForwardVanillaOption**

  Quanto version of a forward vanilla option.

8.48.1 Detailed Description

Quanto version of a forward vanilla option.

8.49 C:/quantlib/QuantLib/ql/instruments/quantovanillaoption.hpp File Reference

Quanto version of a vanilla option.

```
#include <ql/instruments/oneassetoption.hpp>
#include <ql/instruments/payoffs.hpp>
```

Include dependency graph for quantovanillaoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::QuantoOptionResults**< **ResultsType** >

  Results from quanto option calculation
- class **QuantLib::QuantoVanillaOption**

  quanto version of a vanilla option

8.49.1 Detailed Description

Quanto version of a vanilla option.

8.50 C:/quantlib/QuantLib/ql/instruments/stickyratchet.hpp File Reference

Payoffs for double nested options of sticky or ratchet type.

```
#include <ql/option.hpp>
#include <ql/payoff.hpp>
```

Include dependency graph for stickyratchet.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::DoubleStickyRatchetPayoff**
  Intermediate class for single/double sticky/ratchet payoffs.
- class **QuantLib::RatchetPayoff**
  Ratchet payoff (single option)
- class **QuantLib::StickyPayoff**
  Sticky payoff (single option)
- class **QuantLib::RatchetMaxPayoff**
  RatchetMax payoff (double option)
- class **QuantLib::RatchetMinPayoff**
  RatchetMin payoff (double option)
- class **QuantLib::StickyMaxPayoff**
  StickyMax payoff (double option)
- class **QuantLib::StickyMinPayoff**
  StickyMin payoff (double option)

8.50.1 Detailed Description

Payoffs for double nested options of sticky or ratchet type.

8.51 C:/quantlib/QuantLib/ql/instruments/stock.hpp File Reference

concrete stock class

```
#include <ql/instrument.hpp>
#include <ql/quote.hpp>
```

Include dependency graph for stock.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::Stock**
  Simple stock class.

8.51.1 Detailed Description

concrete stock class

8.52 C:/quantlib/QuantLib/ql/instruments/swap.hpp File Reference

Interest rate swap.

```
#include <ql/instrument.hpp>
#include <ql/cashflow.hpp>
```

Include dependency graph for swap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::Swap**
  Interest rate swap.
- class **QuantLib::Swap::arguments**
- class **QuantLib::Swap::results**
- class **QuantLib::Swap::engine**

8.52.1 Detailed Description

Interest rate swap.

8.53 C:/quantlib/QuantLib/ql/instruments/swaption.hpp File Reference

Swaption class.

```
#include <ql/option.hpp>
#include <ql/instruments/vanillaswap.hpp>
#include <ql/termstructures/yieldtermstructure.hpp>
```

Include dependency graph for swaption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- struct **QuantLib::Settlement**
  settlement information
- class **QuantLib::Swaption**
  Swaption class
- class **QuantLib::Swaption::arguments**
  Arguments for swaption calculation
- class **QuantLib::Swaption::engine**
  base class for swaption engines

Functions

- std::ostream & **QuantLib::operator**<< (std::ostream &out, Settlement::Type t)

8.53.1 Detailed Description

Swaption class.

8.54 C:/quantlib/QuantLib/ql/instruments/vanillaoption.hpp File Reference

Vanilla option on a single asset.

```
#include <ql/instruments/oneassetoption.hpp>
#include <ql/instruments/payoffs.hpp>
```

Include dependency graph for vanillaoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::VanillaOption**

  Vanilla option (no discrete dividends, no barriers) on a single asset.

8.54.1 Detailed Description

Vanilla option on a single asset.

8.55 C:/quantlib/QuantLib/ql/instruments/vanillastorageoption.hpp File Reference

vanilla storage option class

```
#include <ql/event.hpp>
#include <ql/exercise.hpp>
#include <ql/instruments/payoffs.hpp>
#include <ql/instruments/oneassetoption.hpp>
```

Include dependency graph for vanillastorageoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::VanillaStorageOption**

  base option class
- class **QuantLib::VanillaStorageOption::arguments**

8.55.1 Detailed Description

vanilla storage option class

8.56 C:/quantlib/QuantLib/ql/instruments/vanillaswap.hpp File Reference

Simple fixed-rate vs Libor swap.

```
#include <ql/instruments/swap.hpp>
#include <ql/time/daycounter.hpp>
#include <ql/time/schedule.hpp>
#include <boost/optional.hpp>
```

Include dependency graph for vanillaswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::VanillaSwap**

  Plain-vanilla swap: fix vs floating leg.
- class **QuantLib::VanillaSwap::arguments**

  Arguments for simple swap calculation
- class **QuantLib::VanillaSwap::results**

  Results from simple swap calculation
- class **QuantLib::VanillaSwap::engine**

Functions

- std::ostream & **QuantLib::operator**<< (std::ostream &out, VanillaSwap::Type t)

8.56.1 Detailed Description

Simple fixed-rate vs Libor swap.

8.57 C:/quantlib/QuantLib/ql/instruments/vanillaswingoption.cpp File Reference

vanilla swing option class

```
#include <ql/event.hpp>
#include <ql/instruments/vanillaswingoption.hpp>
```
Include dependency graph for vanillaswingoption.cpp:

8.57.1 Detailed Description

vanilla swing option class

8.58 C:/quantlib/QuantLib/ql/instruments/vanillaswingoption.hpp File Reference

vanilla swing option class

```
#include <ql/exercise.hpp>
#include <ql/time/daycounter.hpp>
#include <ql/instruments/payoffs.hpp>
#include <ql/instruments/oneassetoption.hpp>
```
Include dependency graph for vanillaswingoption.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::SwingExercise**

  Swing exercise.
- class **QuantLib::VanillaSwingOption**

  base option class
- class **QuantLib::VanillaSwingOption::arguments**

8.58.1 Detailed Description

vanilla swing option class

8.59 C:/quantlib/QuantLib/ql/instruments/varianceswap.hpp File Reference

Variance swap.

```
#include <ql/processes/blackscholesprocess.hpp>
#include <ql/instruments/payoffs.hpp>
#include <ql/option.hpp>
#include <ql/position.hpp>
```

Include dependency graph for varianceswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::VarianceSwap**

  Variance swap.
- class **QuantLib::VarianceSwap::arguments**

  Arguments for forward fair-variance calculation
- class **QuantLib::VarianceSwap::results**

  Results from variance-swap calculation
- class **QuantLib::VarianceSwap::engine**

  base class for variance-swap engines

8.59.1 Detailed Description

Variance swap.

8.60 C:/quantlib/QuantLib/ql/instruments/yearonyearinflationswap.hpp File Reference

Year-on-year inflation-indexed swap.

```
#include <ql/instruments/swap.hpp>
#include <ql/time/calendar.hpp>
#include <ql/time/daycounter.hpp>
#include <ql/time/schedule.hpp>
```

Include dependency graph for yearonyearinflationswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::YearOnYearInflationSwap**

  Year-on-year inflation-indexed swap.
- class **QuantLib::YearOnYearInflationSwap::arguments**

  Arguments for YoY swap calculation
- class **QuantLib::YearOnYearInflationSwap::results**

  Results from YoY swap calculation
- class **QuantLib::YearOnYearInflationSwap::engine**

Functions

- std::ostream & **QuantLib::operator**<< (std::ostream &out, YearOnYearInflationSwap::Type t)

8.60.1 Detailed Description

Year-on-year inflation-indexed swap.

8.61 C:/quantlib/QuantLib/ql/instruments/zerocouponinflationswap.hpp File Reference

Zero-coupon inflation-indexed swap.

```
#include <ql/instruments/swap.hpp>
#include <ql/time/calendar.hpp>
#include <ql/time/daycounter.hpp>
```

Include dependency graph for zerocouponinflationswap.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class **QuantLib::ZeroCouponInflationSwap**

  Zero-coupon inflation-indexed swap.
- class **QuantLib::ZeroCouponInflationSwap::arguments**
- class **QuantLib::ZeroCouponInflationSwap::engine**

8.61.1 Detailed Description

Zero-coupon inflation-indexed swap.

Index