Cloud Computing in QuantLib

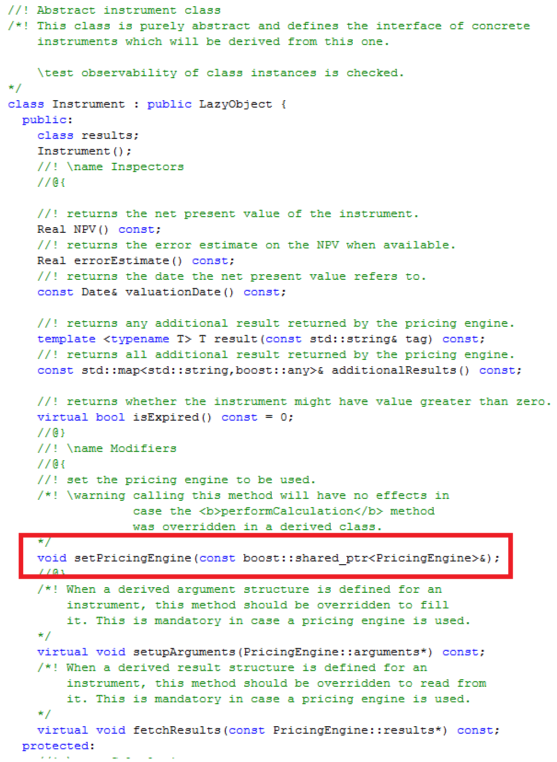
reposit project – status update

=countify – Rate Curve Framework on the cloud

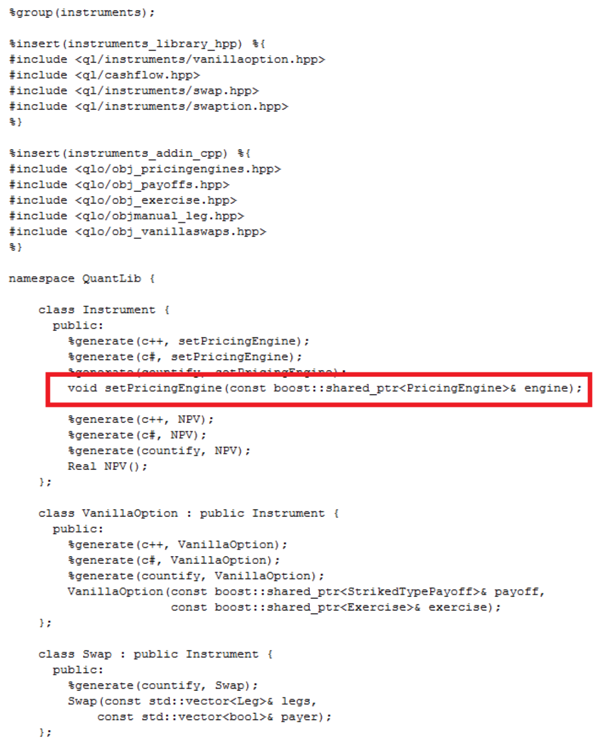
Eric Ehlers
30 November 2015

Reposit is Live

The design is a success and the project meets its goals



```
//! Abstract instrument class
/*! This class is purely abstract and defines the interface of concrete
    instruments which will be derived from this one.

    \test observability of class instances is checked.
*/
class Instrument : public LazyObject {
  public:
    class results;
    Instrument();
    //! \name Inspectors
    //@{

    //! returns the net present value of the instrument.
    Real NPV() const;
    //! returns the error estimate on the NPV when available.
    Real errorEstimate() const;
    //! returns the date the net present value refers to.
    const Date& valuationDate() const;

    //! returns any additional result returned by the pricing engine.
    template <typename T> T result(const std::string& tag) const;
    //! returns all additional result returned by the pricing engine.
    const std::map<std::string,boost::any>& additionalResults() const;

    //! returns whether the instrument might have value greater than zero.
    virtual bool isExpired() const = 0;
    //@}
    //! \name Modifiers
    //@{
    //! set the pricing engine to be used.
    /*! \warning calling this method will have no effects in
                 case the <b>performCalculation</b> method
                 was overridden in a derived class.
    */
    void setPricingEngine(const boost::shared_ptr<PricingEngine>&);
    //@}
    /*! When a derived argument structure is defined for an
        instrument, this method should be overridden to fill
        it. This is mandatory in case a pricing engine is used.
    */
    virtual void setupArguments(PricingEngine::arguments*) const;
    /*! When a derived result structure is defined for an
        instrument, this method should be overridden to read from
        it. This is mandatory in case a pricing engine is used.
    */
    virtual void fetchResults(const PricingEngine::results*) const;
  protected:
```



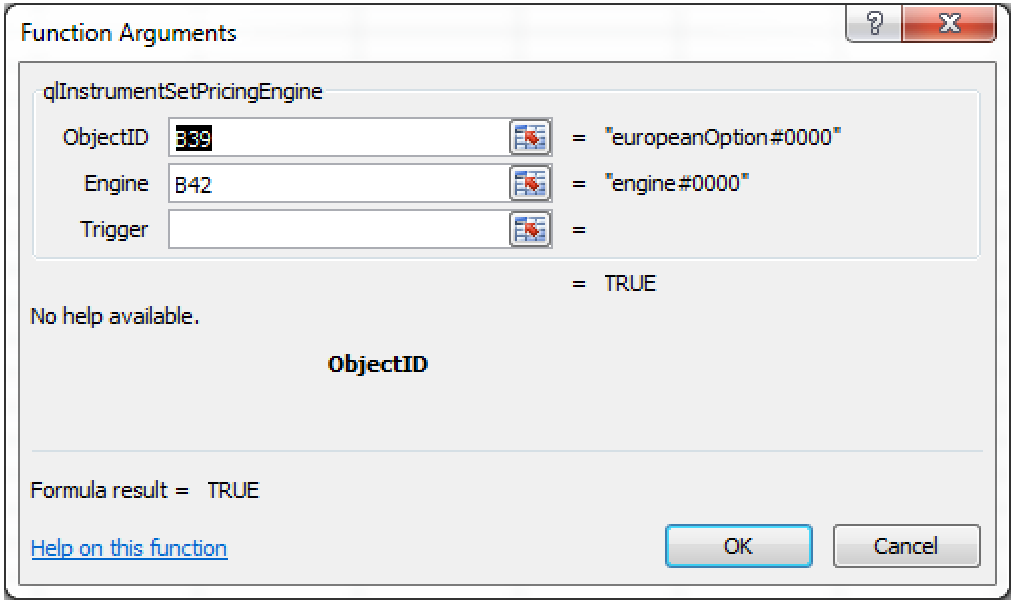
```
%group(instruments);

%insert(instruments_library_hpp) %{
#include <ql/instruments/vanillaoption.hpp>
#include <ql/cashflow.hpp>
#include <ql/instruments/swap.hpp>
#include <ql/instruments/swaption.hpp>
%}

%insert(instruments_addin_cpp) %{
#include <qlo/obj_pricingengines.hpp>
#include <qlo/obj_payoffs.hpp>
#include <qlo/obj_exercise.hpp>
#include <qlo/objmanual_leg.hpp>
#include <qlo/obj_vanillaswaps.hpp>
%}

namespace QuantLib {

    class Instrument {
      public:
        %generate(c++, setPricingEngine);
        %generate(c#, setPricingEngine);
        %generate(countify, setPricingEngine);
        void setPricingEngine(const boost::shared_ptr<PricingEngine>& engine);

        %generate(c++, NPV);
        %generate(c#, NPV);
        %generate(countify, NPV);
        Real NPV();
    };

    class VanillaOption : public Instrument {
      public:
        %generate(c++, VanillaOption);
        %generate(c#, VanillaOption);
        %generate(countify, VanillaOption);
        VanillaOption(const boost::shared_ptr<StrikedTypePayoff>& payoff,
                      const boost::shared_ptr<Exercise>& exercise);
    };

    class Swap : public Instrument {
      public:
        %generate(countify, Swap);
        Swap(const std::vector<Leg>& legs,
             const std::vector<bool>& payer);
    };
```

Exporting Functions - Step One: Copy the function definition from the QuantLib header file to the reposit SWIG interface file

Reposit is Live

The design is a success and the project meets its goals

Exporting Functions - Step Two: Recompile.



qlInstrumentSetPricingEngine

ObjectID B39 = "europeanOption#0000"

Engine B42 = "engine#0000"

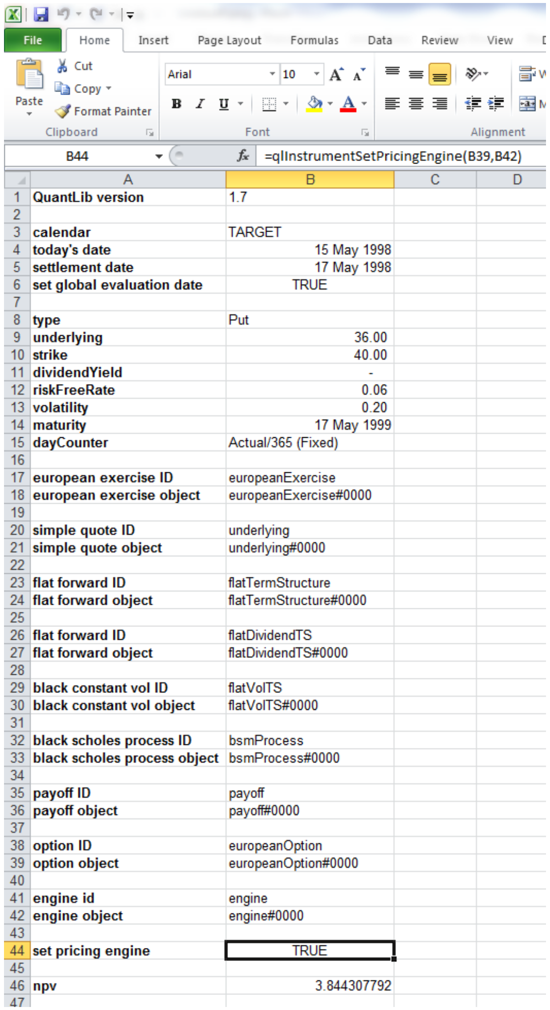
Trigger =

= TRUE

No help available.

ObjectID

Formula result = TRUE

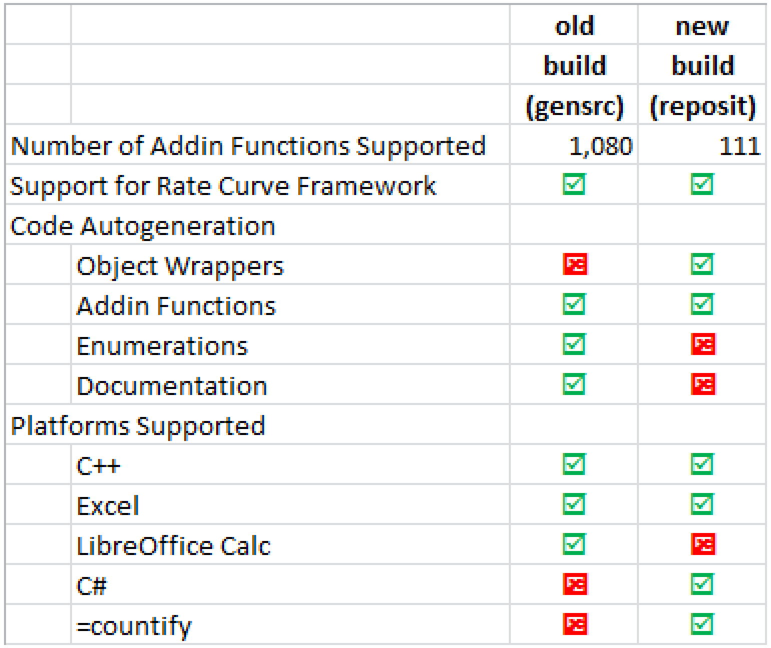


B44 =qlInstrumentSetPricingEngine(B39,B42)

	A	B	C	D
1	QuantLib version	1.7		
2				
3	calendar	TARGET		
4	today's date	15 May 1998		
5	settlement date	17 May 1998		
6	set global evaluation date	TRUE		
7				
8	type	Put		
9	underlying	36.00		
10	strike	40.00		
11	dividendYield	-		
12	riskFreeRate	0.06		
13	volatility	0.20		
14	maturity	17 May 1999		
15	dayCounter	Actual/365 (Fixed)		
16				
17	european exercise ID	europeanExercise		
18	european exercise object	europeanExercise#0000		
19				
20	simple quote ID	underlying		
21	simple quote object	underlying#0000		
22				
23	flat forward ID	flatTermStructure		
24	flat forward object	flatTermStructure#0000		
25				
26	flat forward ID	flatDividendTS		
27	flat forward object	flatDividendTS#0000		
28				
29	black constant vol ID	flatVolTS		
30	black constant vol object	flatVolTS#0000		
31				
32	black scholes process ID	bsmProcess		
33	black scholes process object	bsmProcess#0000		
34				
35	payoff ID	payoff		
36	payoff object	payoff#0000		
37				
38	option ID	europeanOption		
39	option object	europeanOption#0000		
40				
41	engine id	engine		
42	engine object	engine#0000		
43				
44	set pricing engine	TRUE		
45				
46	npv	3.844307792		

Project Status

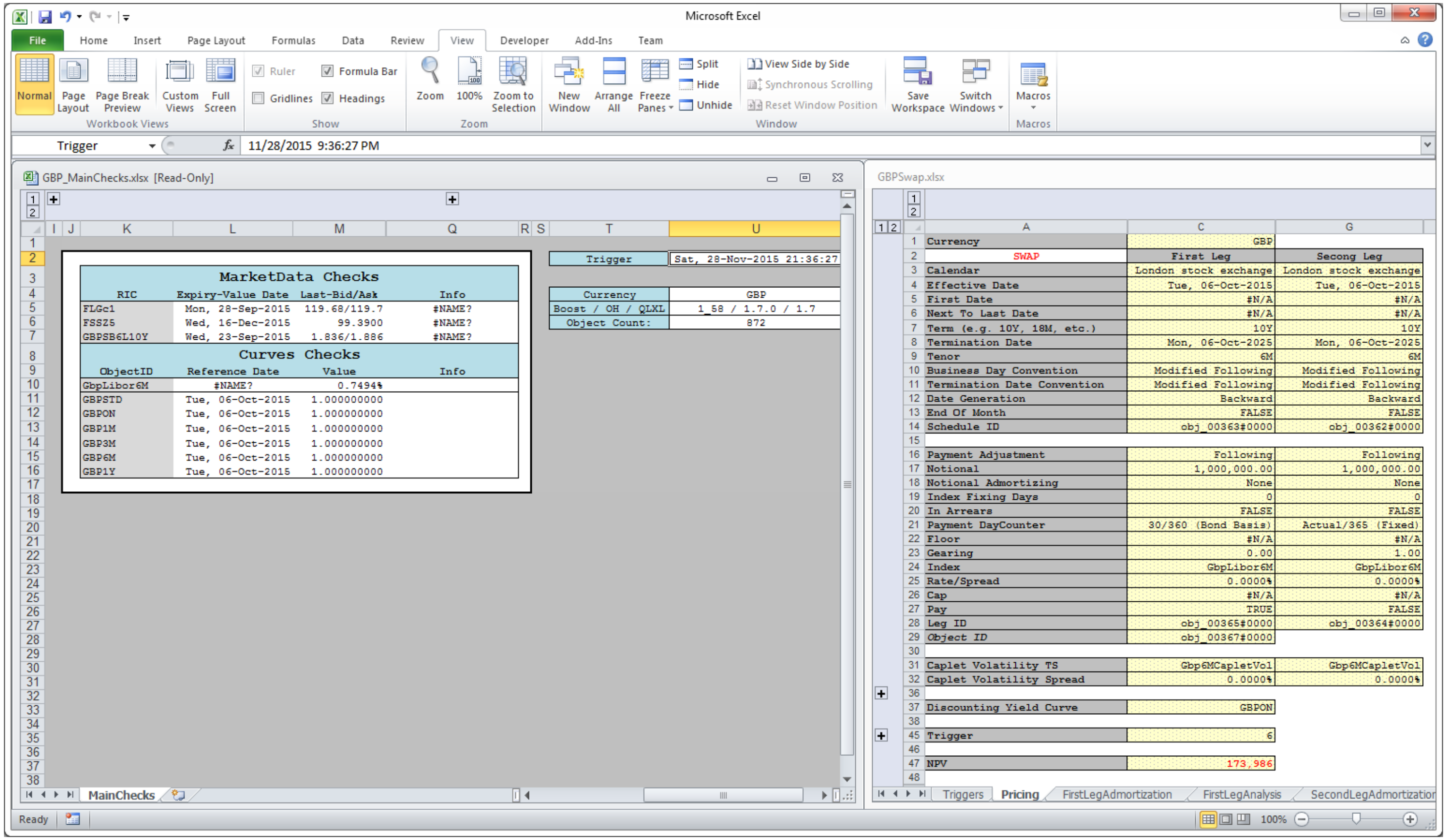
Version 1.7



		old	new
		build	build
		(gensrc)	(reposit)
Number of Addin Functions Supported		1,080	111
Support for Rate Curve Framework		☑	☑
Code Autogeneration			
	Object Wrappers	☒	☑
	Addin Functions	☑	☑
	Enumerations	☑	☒
	Documentation	☑	☒
Platforms Supported			
	C++	☑	☑
	Excel	☑	☑
	LibreOffice Calc	☑	☒
	C#	☒	☑
	=countify	☒	☑

=countify

reposit on the cloud



Trigger | 11/28/2015 9:36:27 PM

GBP_MainChecks.xlsx [Read-Only]

MarketData Checks			
RIC	Expiry-Value Date	Last-Bid/Ask	Info
FLGc1	Mon, 28-Sep-2015	119.68/119.7	#NAME?
FSSZ5	Wed, 16-Dec-2015	99.3900	#NAME?
GBPSB6L10Y	Wed, 23-Sep-2015	1.836/1.886	#NAME?

Curves Checks			
ObjectID	Reference Date	Value	Info
GbpLibor6M	#NAME?	0.7494%	
GBPSTD	Tue, 06-Oct-2015	1.000000000	
GBPON	Tue, 06-Oct-2015	1.000000000	
GBP1M	Tue, 06-Oct-2015	1.000000000	
GBP3M	Tue, 06-Oct-2015	1.000000000	
GBP6M	Tue, 06-Oct-2015	1.000000000	
GBP1Y	Tue, 06-Oct-2015	1.000000000	

Trigger	Sat, 28-Nov-2015 21:36:27

Currency	GBP
Boost / OH / QLXL	1_58 / 1.7.0 / 1.7
Object Count:	872

GBPSwap.xlsx

	A	C	G
1	Currency	GBP	
2	SWAP	First Leg	Secong Leg
3	Calendar	London stock exchange	London stock exchange
4	Effective Date	Tue, 06-Oct-2015	Tue, 06-Oct-2015
5	First Date	#N/A	#N/A
6	Next To Last Date	#N/A	#N/A
7	Term (e.g. 10Y, 18M, etc.)	10Y	10Y
8	Termination Date	Mon, 06-Oct-2025	Mon, 06-Oct-2025
9	Tenor	6M	6M
10	Business Day Convention	Modified Following	Modified Following
11	Termination Date Convention	Modified Following	Modified Following
12	Date Generation	Backward	Backward
13	End Of Month	FALSE	FALSE
14	Schedule ID	obj_00363#0000	obj_00362#0000
15			
16	Payment Adjustment	Following	Following
17	Notional	1,000,000.00	1,000,000.00
18	Notional Admortizing	None	None
19	Index Fixing Days	0	0
20	In Arrears	FALSE	FALSE
21	Payment DayCounter	30/360 (Bond Basis)	Actual/365 (Fixed)
22	Floor	#N/A	#N/A
23	Gearing	0.00	1.00
24	Index	GbpLibor6M	GbpLibor6M
25	Rate/Spread	0.0000%	0.0000%
26	Cap	#N/A	#N/A
27	Pay	TRUE	FALSE
28	Leg ID	obj_00365#0000	obj_00364#0000
29	*Object ID*	obj_00367#0000	
30			
31	Caplet Volatility TS	Gbp6MCapletVol	Gbp6MCapletVol
32	Caplet Volatility Spread	0.0000%	0.0000%
36			
37	Discounting Yield Curve	GBPON	
38			
45	Trigger	6	
46			
47	NPV	173,986	
48			

Bootstrap the yield curves

Price a trade

The Rate Curve Framework has been deployed to the =countify platform.