```
   Help
#include "local_vol_callable.h"
#include "enums.h"
#include "pnl/pnl_random.h"
#include "pnl/pnl_basis.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2011+2) //The "#else" part of the code will be freely av
    ailable after the (year of creation of this file + 2)

int CALC(MC_RIBSDE)(void *Opt,void *Mod,PricingMethod *Met)
{
  return AVAILABLE_IN_FULL_PREMIA;
}

static int CHK_OPT(MC_RIBSDE)(void *Opt, void *Mod)
{
  return NONACTIVE;
}

static int MET(Init)(PricingMethod *Met,Option *Opt)
{
  if ( Met->init == 0)
    {
      Met->init=1;
    }

  return OK;
}

#else

static int(*price_func)(double *prix, TYPEMOD *Mod, TYPE
    OPT *Opt, int gen, int bindex, int m, int M, int steps) = NUL
    L;

extern int callable_highly_path_dep_protection(double *prix
    , TYPEMOD *Mod, TYPEOPT *Opt, int gen, int bindex, int m,
    int M, int steps);
extern int callable_intermittent_protection(double *prix,
    TYPEMOD *Mod, TYPEOPT *Opt, int gen, int bindex, int m,
    int M, int steps);
extern int callable_no_call_protection(double *prix, TYPE
    MOD *Mod, TYPEOPT *Opt, int gen, int bindex, int m, int M,
    int steps);
extern int callable_path_dep_proctection(double *prix, TYPE
    MOD *Mod, TYPEOPT *Opt, int gen, int bindex, int m, int M,
    int steps);
extern int callable_std_protection (double *prix, TYPEMOD *
    Mod, TYPEOPT *Opt, int gen, int bindex, int m, int M, int
    steps);

int CALC(MC_RIBSDE)(void *Opt,void *Mod,PricingMethod *Met)
{
  TYPEOPT* ptOpt=(TYPEOPT*)Opt;
  TYPEMOD* ptMod=(TYPEMOD*)Mod;

  if (price_func == NULL)
    {
      printf("Some initialization is missing");
      return FAIL;
    }
  (*price_func)(&(Met->Res[0].Val.V_DOUBLE), ptMod, ptOpt,
    Met->Par[1].Val.V_ENUM.value, Met->Par[2].Val.V_ENUM.value,
    Met->Par[3].Val.V_INT, Met->Par[0].Val.V_LONG, Met->Par[4].
    Val.V_INT);
  return OK;
}

static int CHK_OPT(MC_RIBSDE)(void *Opt, void *Mod)
{
  return OK;
}

static int MET(Init)(PricingMethod *Met,Option *Opt)
{
  if ( Met->init == 0)
    {
      Met->Par[0].Val.V_LONG = 10000;
      Met->Par[1].Val.V_ENUM.value=0;
      Met->Par[1].Val.V_ENUM.members=&PremiaEnumMCRNGs;
      Met->Par[2].Val.V_ENUM.value=0;
      Met->Par[2].Val.V_ENUM.members=&PremiaEnumBasis;
      Met->Par[3].Val.V_INT=4;
      Met->Par[4].Val.V_INT=4;

      Met->init=1;
    }

  if (Opt)
  {
    if (strcmp(Opt->Name, "NoCall") == 0) price_func =     callable_no_call_prot
    if (strcmp(Opt->Name, "StdCall") == 0) price_func =     callable_std_protect
    if (strcmp(Opt->Name, "PathDepCall") == 0) price_func
    = callable_path_dep_proctection;
    if (strcmp(Opt->Name, "HighlyPathDepCall") == 0)
    price_func = callable_highly_path_dep_protection;
    if (strcmp(Opt->Name, "IntermittentCall") == 0)
    price_func = callable_intermittent_protection;
  }

  return OK;
}

#endif //PremiaCurrentVersion

PricingMethod MET(MC_RIBSDE)=
{
  "MC_RIBSDE",
  {{"Nb iterations",LONG,{1000},ALLOW},
   {"RandomGenerator",ENUM,{0},ALLOW},
   {"Basis",ENUM,{1},ALLOW},
   {"Basis Dimension",INT,{100},ALLOW},
   {"Nb discretization steps per day", INT, {4}, ALLOW},
   {" ",PREMIA_NULLTYPE,{0},FORBID}},
  CALC(MC_RIBSDE),
  {{"Price",DOUBLE,{100},FORBID},{" ",PREMIA_NULLTYPE,{0},
    FORBID}},
  CHK_OPT(MC_RIBSDE),
  CHK_ok,
  MET(Init)
} ;
```

References