```
   Help
#include  "doublehes1d_vol.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2010+2) //The "#else" part of the code will be freely av
    ailable after the (year of creation of this file + 2)
static int CHK_OPT(CF_VarSwapDoubleHeston)(void *Opt, void
    *Mod)
{
  return NONACTIVE;
}
int CALC(CF_VarSwapDoubleHeston)(void*Opt,void *Mod,Pricing
    Method *Met)
{
return AVAILABLE_IN_FULL_PREMIA;
}
#else

int VarSwapDoubleHeston(double S,double T, double strike,
    double r, double divid,
            double z1, double z2,double z3,
            double k,double c,double sigma1,double si
    gma2,double rho1,double rho2,double rho3,
            double *fairval, double *ptprice)
{
  double res;

    if (k==c)
  {
  res=(2.*z3-z2)*T + ((z1+z2-2.*z3)/k + (z2-z3)*T)*(1.0 -
    exp(-k*T));
  }
    else
  {
    res= z3*T + ((z1-z3)/k - (z2-z3)/(k-c))*(1.0 - exp(-k*
    T)) + (z2-z3)*k/c/(k - c)*(1.0 - exp(-c*T));
  }
  *fairval = sqrt(res/T)*100.;
  *ptprice= exp(-r*T)*(res*10000.0/T-strike*strike);
  /* Price*/
  return OK;
}

int CALC(CF_VarSwapDoubleHeston)(void *Opt, void *Mod,
    PricingMethod *Met)
{
  TYPEOPT* ptOpt=(TYPEOPT*)Opt;
  TYPEMOD* ptMod=(TYPEMOD*)Mod;
  double r,divid, strike;
  NumFunc_1 *p;

      r=log(1.+ptMod->R.Val.V_DOUBLE/100.);
      divid=log(1.+ptMod->Divid.Val.V_DOUBLE/100.);
  p=ptOpt->PayOff.Val.V_NUMFUNC_1;
     strike=p->Par[0].Val.V_DOUBLE;

      return VarSwapDoubleHeston(ptMod->S0.Val.V_PDOUBLE,
                                 ptOpt->Maturity.Val.V_DA
    TE-ptMod->T.Val.V_DATE,
                                 strike, r,
                                 divid, ptMod->Sigma0.Val.
    V_PDOUBLE,
                                 ptMod->Sigma0V.Val.V_PDOUB
    LE,
                                 ptMod->LongRunVarianceV.
    Val.V_PDOUBLE
                                 ,ptMod->MeanReversion.hal.
    V_PDOUBLE
                                 ,ptMod->MeanReversionV.Val
    .V_PDOUBLE,
                                 ptMod->Sigma.Val.V_PDOUB
    LE,
                                 ptMod->SigmaV.Val.V_PDOUB
    LE,
                                 ptMod->Rho.Val.V_DOUBLE,
                                 ptMod->RhoSV2.Val.V_
    DOUBLE,
                                 ptMod->RhoVV.Val.V_DOUBLE,
                                 &(Met->Res[0].Val.V_
    DOUBLE),
                 &(Met->Res[1].Val.V_DOUBLE)
```

```
        );

}


static int CHK_OPT(CF_VarSwapDoubleHeston)(void *Opt, void
    *Mod)
{
  if ((strcmp( ((Option*)Opt)->Name,"VarianceSwap")==0 ))
      return OK;

  return  WRONG;
}
#endif //PremiaCurrentVersion

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

  return OK;
}

PricingMethod MET(CF_VarSwapDoubleHeston)=
{
  "CF_VarianceSwap_DHes",
  {{" ",PREMIA_NULLTYPE,{0},FORBID}},
  CALC(CF_VarSwapDoubleHeston),
  {{"Fair strike in annual volatility points",DOUBLE,{100},
    FORBID},
   {"Price, in 10000 variance points",DOUBLE,{100},FORBID},
   {" ",PREMIA_NULLTYPE,{0},FORBID}},
  CHK_OPT(CF_VarSwapDoubleHeston),
  CHK_ok,
  MET(Init)
};
```

References