KTAB

Generated by Doxygen 1.8.11

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 DemoLeon Namespace Reference

Classes

- class LeonActor
- class LeonModel
- class LeonState

Functions

- LeonModel * demoSetup (unsigned int numFctr, unsigned int numCGrp, unsigned int numSect, uint64_t s, PRNG *rng)
- void demoEUEcon (uint64_t s, unsigned int numF, unsigned int numG, unsigned int numS, PRNG *rng)
- void demoMaxEcon (uint64_t s, unsigned int numF, unsigned int numG, unsigned int numS, PRNG *rng)
- void demoEUEcon (uint64_t s, PRNG *rng)
- void demoMaxEcon (uint64_t s, PRNG *rng)

Variables

- const double TolIFD = 1E-6

5.1.1 Function Documentation

5.1.1.1 void DemoLeon::demoEUEcon (uint64_t *s*, PRNG * *rng*)

5.1.1.2 void DemoLeon::demoEUEcon (uint64_t *s*, unsigned int *numF*, unsigned int *numG*, unsigned int *numS*, PRNG * *rng*)

5.1.1.3 void DemoLeon::demoMaxEcon (uint64_t *s*, PRNG * *rng*)

5.1.1.4 void DemoLeon::demoMaxEcon (uint64_t *s*, unsigned int *numF*, unsigned int *numG*, unsigned int *numS*, PRNG * *rng*)

5.1.1.5 LeonModel * DemoLeon::demoSetup (unsigned int *numFctr*, unsigned int *numCGrp*, unsigned int *numSect*, uint64_t *s*, PRNG * *rng*)

5.1.2 Variable Documentation

5.1.2.1 const double DemoLeon::TolIFD = 1E-6

5.2 DemoMtch Namespace Reference

Classes

- class MtchActor
- class MtchModel
- class MtchState

Functions

- bool equivMtchPstn (const MtchPstn &mp1, const MtchPstn &mp2)
- void showMtchPstn (const MtchPstn &mp)
- bool stableMtchState (unsigned int iter, const State *s1)
- void demoDivideSweets (uint64_t s, PRNG *rng)
- void demoMaxSupport (uint64_t s, PRNG *rng)
- void demoMtchSUSN (uint64_t s, PRNG *rng)
- void multiMtchSUSN (uint64_t s, PRNG *rng)
- bool oneMtchSUSN (uint64_t s, PRNG *rng)

5.2.1 Function Documentation

5.2.1.1 void DemoMtch::demoDivideSweets (uint64_t *s*, PRNG * *rng*)

5.2.1.2 void DemoMtch::demoMaxSupport (uint64_t *s*, PRNG * *rng*)

5.2.1.3 void DemoMtch::demoMtchSUSN (uint64_t *s*, PRNG * *rng*)

5.2.1.4 bool DemoMtch::equivMtchPstn (const MtchPstn & *mp1*, const MtchPstn & *mp2*)

5.2.1.5 void DemoMtch::multiMtchSUSN (uint64_t *s*, PRNG * *rng*)

5.2.1.6 bool DemoMtch::oneMtchSUSN (uint64_t *s*, PRNG * *rng*)

5.2.1.7 void DemoMtch::showMtchPstn (const MtchPstn & *mp*)

5.2.1.8 bool DemoMtch::stableMtchState (unsigned int *iter*, const State * *s1*)

5.3 KBase Namespace Reference

Classes

- class Actor
- class GAOpt
- class GHCSearch
- class KException
- class KMatrix
- class Model
- class MtchGene
- class MtchPstn
- class Position
- class PRNG

- class State
- class VctrPstn
- class VHCSearch

Enumerations

- enum VotingRule : char {
  VotingRule::Binary, VotingRule::PropBin, VotingRule::Proportional, VotingRule::PropCbc,
  VotingRule::Cubic }
- enum ThirdPartyCommit { ThirdPartyCommit::NoCommit, ThirdPartyCommit::SemiCommit, ThirdPartyCommit::FullCommit }
- enum ReportingLevel {
  ReportingLevel::Silent, ReportingLevel::Low, ReportingLevel::Medium, ReportingLevel::High,
  ReportingLevel::Debugging }

Functions

- string vrName (VotingRule vr)
- string tpcName (ThirdPartyCommit tpc)
- vector< MtchPstn > uniqueMP (vector< MtchPstn > mps)
- unsigned int crossSite (PRNG *rng, unsigned int nc)
- KMatrix trans (const KMatrix &m1)
- double norm (const KMatrix &m1)
- double sum (const KMatrix &m1)
- double mean (const KMatrix &m1)
- double stdv (const KMatrix &m1)
- double maxAbs (const KMatrix &m)
- tuple< unsigned int, unsigned int > ndxMaxAbs (const KMatrix &m)
- double lCorr (const KMatrix &m1, const KMatrix &m2)
- double dot (const KMatrix &m1, const KMatrix &m2)
- KMatrix operator+ (const KMatrix &m1, double x)
- KMatrix operator- (const KMatrix &m1, double x)
- bool sameShape (const KMatrix &m1, const KMatrix &m2)
- KMatrix operator+ (const KMatrix &m1, const KMatrix &m2)
- KMatrix operator- (const KMatrix &m1, const KMatrix &m2)
- KMatrix operator* (double x, const KMatrix &m1)
- KMatrix operator/ (const KMatrix &m1, double x)
- KMatrix operator* (const KMatrix &m1, const KMatrix &m2)
- KMatrix inv (const KMatrix &m)
- KMatrix iMat (unsigned int n)
- KMatrix makePerp (const KMatrix &x, const KMatrix &p)
- KMatrix joinH (const KMatrix &mL, const KMatrix &mR)
- KMatrix joinV (const KMatrix &mT, const KMatrix &mB)
- double sqr (const double x)
- double qrtc (const double x)
- std::chrono::time_point< std::chrono::system_clock > displayProgramStart ()
- void displayProgramEnd (std::chrono::time_point< std::chrono::system_clock > st)
- char * newChars (unsigned int len)
- double rescale (double x, double x0, double x1, double y0, double y1)
- template<typename T >
  T popBack (vector< T > &v)
- uint64_t qTrans (uint64_t s)
- KMatrix projPos (const KMatrix &w)
- KMatrix projBox (const KMatrix &lb, const KMatrix &ub, const KMatrix &w)

- tuple< KMatrix, unsigned int, KMatrix > viABG (const KMatrix &xInit, function< KMatrix(const KMatrix &x)> F, function< KMatrix(const KMatrix &x)> P, double beta, double thresh, unsigned int iMax, bool extra)
- tuple< KMatrix, unsigned int, KMatrix > viBSHe96 (const KMatrix &M, const KMatrix &q, function< KMatrix(const KMatrix &)> pK, KMatrix u0, const double eps, const unsigned int iMax)
- tuple< KMatrix, KMatrix, KMatrix, KMatrix > antiLemke (unsigned int n)

5.3.1 Enumeration Type Documentation

5.3.1.1 enum KBase::ReportingLevel [strong]

Enumerator

Silent

Low

Medium

High

Debugging

5.3.1.2 enum KBase::ThirdPartyCommit [strong]

Enumerator

NoCommit

SemiCommit

FullCommit

5.3.1.3 enum KBase::VotingRule : char [strong]

Enumerator

Binary

PropBin

Proportional

PropCbc

Cubic

5.3.2 Function Documentation

5.3.2.1 tuple<KMatrix, KMatrix, KMatrix, KMatrix> KBase::antiLemke (unsigned int *n*)

5.3.2.2 unsigned int KBase::crossSite (PRNG * *rng*, unsigned int *nc*)

5.3.2.3 void KBase::displayProgramEnd (std::chrono::time_point< std::chrono::system_clock > *st*)

5.3.2.4 std::chrono::time_point< std::chrono::system_clock > KBase::displayProgramStart ( )

5.3.2.5 double KBase::dot (const KMatrix & *m1*, const KMatrix & *m2*)

5.3.2.6 KMatrix KBase::iMat (unsigned int *n*)

5.3.2.7 KMatrix KBase::inv (const KMatrix & *m*)

5.3.2.8 KMatrix KBase::joinH (const KMatrix & *mL,* const KMatrix & *mR*)

5.3.2.9 KMatrix KBase::joinV (const KMatrix & *mT,* const KMatrix & *mB*)

5.3.2.10 double KBase::lCorr (const KMatrix & *m1,* const KMatrix & *m2*)

5.3.2.11 KMatrix KBase::makePerp (const KMatrix & *x,* const KMatrix & *p*)

5.3.2.12 double KBase::maxAbs (const KMatrix & *m*)

5.3.2.13 double KBase::mean (const KMatrix & *m1*)

5.3.2.14 tuple< unsigned int, unsigned int > KBase::ndxMaxAbs (const KMatrix & *m*)

5.3.2.15 char * KBase::newChars (unsigned int *len*)

5.3.2.16 double KBase::norm (const KMatrix & *m1*)

5.3.2.17 KMatrix KBase::operator* (double *x,* const KMatrix & *m1*)

5.3.2.18 KMatrix KBase::operator* (const KMatrix & *m1,* const KMatrix & *m2*)

5.3.2.19 KMatrix KBase::operator+ (const KMatrix & *m1,* double *x*)

5.3.2.20 KMatrix KBase::operator+ (const KMatrix & *m1,* const KMatrix & *m2*)

5.3.2.21 KMatrix KBase::operator- (const KMatrix & *m1,* double *x*)

5.3.2.22 KMatrix KBase::operator- (const KMatrix & *m1,* const KMatrix & *m2*)

5.3.2.23 KMatrix KBase::operator/ (const KMatrix & *m1,* double *x*)

5.3.2.24 template<typename T > T KBase::popBack (vector< T > & *v*)

5.3.2.25 KMatrix KBase::projBox (const KMatrix & *lb,* const KMatrix & *ub,* const KMatrix & *w*)

5.3.2.26 KMatrix KBase::projPos (const KMatrix & *w*)

5.3.2.27 double KBase::qrtc (const double *x*)

5.3.2.28 uint64_t KBase::qTrans (uint64_t *s*)

5.3.2.29 double KBase::rescale (double *x,* double *x0,* double *x1,* double *y0,* double *y1*)

5.3.2.30 bool KBase::sameShape (const KMatrix & *m1,* const KMatrix & *m2*)

5.3.2.31 double KBase::sqr (const double *x*)

5.3.2.32 double KBase::stdv (const KMatrix & *m1*)

5.3.2.33 double KBase::sum (const KMatrix & *m1*)

5.3.2.34 string KBase::tpcName (ThirdPartyCommit *tpc*)

5.3.2.35 KMatrix KBase::trans (const KMatrix & *m1*)

5.3.2.36 vector< MtchPstn > KBase::uniqueMP (vector< MtchPstn > *mps*)

5.3.2.37 tuple< KMatrix, unsigned int, KMatrix > KBase::viABG (const KMatrix & *xInit*, function< KMatrix(const KMatrix &x)> *F*, function< KMatrix(const KMatrix &x)> *P*, double *beta*, double *thresh*, unsigned int *iMax*, bool *extra*)

5.3.2.38 tuple< KMatrix, unsigned int, KMatrix > KBase::viBSHe96 (const KMatrix & *M*, const KMatrix & *q*, function< KMatrix(const KMatrix &)> *pK*, KMatrix *u0*, const double *eps*, const unsigned int *iMax*)

5.3.2.39 string KBase::vrName (VotingRule *vr*)

5.4 KGraph Namespace Reference

Classes

- class Canvas
- class CoordMap
- class Picture

5.5 MDemo Namespace Reference

Classes

- class SQLDB
- class ZActor

Functions

- void demoPCE (uint64_t s, PRNG *rng)
- void demoSpVSR (uint64_t s, PRNG *rng)
- void demoDBObject ()

5.5.1 Function Documentation

5.5.1.1 void MDemo::demoDBObject ( )

5.5.1.2 void MDemo::demoPCE (uint64_t *s*, PRNG * *rng*)

5.5.1.3 void MDemo::demoSpVSR (uint64_t *s*, PRNG * *rng*)

5.6 Tetris Namespace Reference

Classes

- class Board
- class ControlState
- class PVCanvas
- class Shape
- class TApp
- class TCanvas

Enumerations

- enum TCode {
  N = 0, I, J, L,
  O, S, T, Z }
- enum SchemeShapes {
  SS_GameBoy, SS_Gerasimov, SS_Sega, SS_SovietMG,
  SS_TetrisCo }
- enum SchemeWindows { SW_Black, SW_White, SW_Beige }

Functions

- char * newChar (unsigned int buffLen)
- void tetrisTimer (void *)

5.6.1 Enumeration Type Documentation

5.6.1.1 enum Tetris::SchemeShapes

Enumerator

SS_GameBoy

SS_Gerasimov

SS_Sega

SS_SovietMG

SS_TetrisCo

5.6.1.2 enum Tetris::SchemeWindows

Enumerator

SW_Black

SW_White

SW_Beige

5.6.1.3 enum Tetris::TCode

Enumerator

N

I

J

L

O

S

T

Z

5.6.2 Function Documentation

5.6.2.1 char * Tetris::newChar (unsigned int *buffLen*)

5.6.2.2 void Tetris::tetrisTimer (void *)

5.7 UDemo Namespace Reference

Classes

- class TargetedBV

Typedefs

- typedef vector< bool > BVec

Functions

- void show (string str, const KMatrix &m, string fs)
- double nProd (double x, double y)
- double bsu (double d, double R)
- double bvu (const KBase::KMatrix &d, const KBase::KMatrix &s, double R)
- void demoThreadLambda (unsigned int n)
- void demoThreadSynch (unsigned int n)
- void demoMatrix (PRNG *rng)
- void demoABG00 (PRNG *rng)
- double eNorm (const KMatrix &a, const KMatrix &x)
- KMatrix eUnitize (const KMatrix &a, const KMatrix &x)
- KMatrix projEllipse (const KMatrix &a, const KMatrix &w)
- void demoEllipseLVI (PRNG *rng, unsigned int n)
- tuple< KMatrix, KMatrix, KMatrix, KMatrix > antiLemke (unsigned int n)
- void demoAntiLemke (PRNG *rng, unsigned int n)
- void demoEllipse (PRNG *rng)
- void demoGA (PRNG *rng)
- void demoGHC (PRNG *rng)
- void demoVHC00 (PRNG *rng)
- void demoVHC01 (PRNG *rng)
- void demoVHC02 (PRNG *rng)
- void demoVHC03 (PRNG *rng)
- void parallelMatrixMult (PRNG *rng)

5.7.1 Typedef Documentation

5.7.1.1 typedef vector<bool> UDemo::BVec

5.7.2 Function Documentation

5.7.2.1 tuple<KMatrix, KMatrix, KMatrix, KMatrix> UDemo::antiLemke (unsigned int *n*)

5.7.2.2 double UDemo::bsu (double *d*, double *R*)

5.7.2.3 double UDemo::bvu (const KBase::KMatrix & *d*, const KBase::KMatrix & *s*, double *R*)

5.7.2.4 void UDemo::demoABG00 (PRNG * *rng*)

5.7.2.5 void UDemo::demoAntiLemke (PRNG * *rng,* unsigned int *n*)

5.7.2.6 void UDemo::demoEllipse (PRNG * *rng*)

5.7.2.7 void UDemo::demoEllipseLVI (PRNG * *rng,* unsigned int *n*)

5.7.2.8 void UDemo::demoGA (PRNG * *rng*)

5.7.2.9 void UDemo::demoGHC (PRNG * *rng*)

5.7.2.10 void UDemo::demoMatrix (PRNG * *rng*)

5.7.2.11 void UDemo::demoThreadLambda (unsigned int *n*)

5.7.2.12 void UDemo::demoThreadSynch (unsigned int *n*)

5.7.2.13 void UDemo::demoVHC00 (PRNG * *rng*)

5.7.2.14 void UDemo::demoVHC01 (PRNG * *rng*)

5.7.2.15 void UDemo::demoVHC02 (PRNG * *rng*)

5.7.2.16 void UDemo::demoVHC03 (PRNG * *rng*)

5.7.2.17 double UDemo::eNorm (const KMatrix & *a,* const KMatrix & *x*)

5.7.2.18 KMatrix UDemo::eUnitize (const KMatrix & *a,* const KMatrix & *x*)

5.7.2.19 double UDemo::nProd (double *x,* double *y*)

5.7.2.20 void UDemo::parallelMatrixMult (PRNG * *rng*)

5.7.2.21 KMatrix UDemo::projEllipse (const KMatrix & *a,* const KMatrix & *w*)

5.7.2.22 void UDemo::show (string *str,* const KMatrix & *m,* string *fs*)

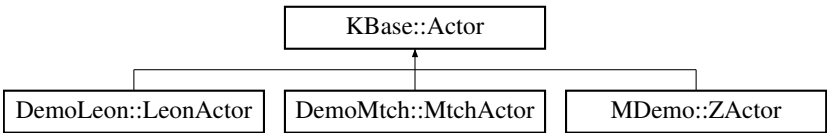
Chapter 6

Class Documentation

6.1 KBase::Actor Class Reference

`#include <kmodel.h>`

Inheritance diagram for KBase::Actor:



Public Member Functions

- Actor (string n, string d)
- virtual ~Actor ()
- virtual double vote (unsigned int p1, unsigned int p2, const State *st) const =0

Static Public Member Functions

- static double thirdPartyVoteSU (double wk, VotingRule vr, ThirdPartyCommit comm, double pik, double pjk, double uki, double ukj, double ukk)
- static double vProbLittle (VotingRule vr, double wn, double uni, double unj, double contrib_i_ij, double contrib_j_ij)

Public Attributes

- string name = "GA"
- string desc = "Generic Actor"

6.1.1 Constructor & Destructor Documentation

6.1.1.1 KBase::Actor::Actor (string *n*, string *d*)

6.1.1.2 KBase::Actor::~Actor ( ) [virtual]

6.1.2 Member Function Documentation

6.1.2.1 double KBase::Actor::thirdPartyVoteSU (double *wk,* VotingRule *vr,* ThirdPartyCommit *comm,* double *pik,* double *pjk,* double *uki,* double *ukj,* double *ukk*) [static]

6.1.2.2 virtual double KBase::Actor::vote (unsigned int *p1,* unsigned int *p2,* const State * *st*) const [pure virtual]

Implemented in DemoMtch::MtchActor, DemoLeon::LeonActor, and MDemo::ZActor.

6.1.2.3 double KBase::Actor::vProbLittle (VotingRule *vr,* double *wn,* double *uni,* double *unj,* double *contrib_i_ij,* double *contrib_j_ij*) [static]

6.1.3 Member Data Documentation

6.1.3.1 string KBase::Actor::desc = "Generic Actor"

6.1.3.2 string KBase::Actor::name = "GA"

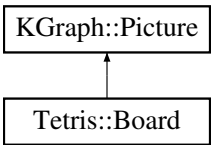
The documentation for this class was generated from the following files:

- kmodel.h
- kmodel.cpp

6.2 Tetris::Board Class Reference

`#include <board.h>`

Inheritance diagram for Tetris::Board:



Public Member Functions

- Board (unsigned int r, unsigned int c)
- virtual ~Board ()
- void randomizeFragments (double f)
- void rotateDown ()
- virtual void update (Canvas *c) const
- unsigned int nFromIJ (int i, int j) const
- bool resetCurrPiece ()
- unsigned int clearLines ()
- unsigned int stepGame ()
- bool tryLRot ()
- bool tryRRot ()
- bool tryLMove ()
- bool tryRMove ()
- bool testSDrop ()
- bool trySDrop ()
- bool tryHDrop ()
- void drawShape (int i, int j, Canvas *cnvs) const
- void drawUnitSquare (Fl_Color clr1, int i, int j, bool dotP, Fl_Color clr2, Canvas *cnvs) const

Public Attributes

- unsigned int rows = 0
- unsigned int clms = 0
- Shape currShape = Shape()
- int currI = 0
- int currJ = 0
- Shape nextShape = Shape()

Protected Member Functions

- void drawBackground (Canvas *c) const
- void drawCurrShape (Canvas *c) const
- void drawFragments (Canvas *c) const
- bool testShape (Shape s, int i, int j) const
- void randomizeRow (unsigned int i)
- vector< TCode > emptyBoard () const
- void placeShape (Shape s, int i, int j)
- bool clearOneLine (const unsigned int i)

Additional Inherited Members

6.2.1 Constructor & Destructor Documentation

6.2.1.1 Tetris::Board::Board (unsigned int *r*, unsigned int *c*)

6.2.1.2 Tetris::Board::~Board ( ) [virtual]

6.2.2 Member Function Documentation

6.2.2.1 unsigned int Tetris::Board::clearLines ( )

6.2.2.2 bool Tetris::Board::clearOneLine (const unsigned int *i*) [protected]

6.2.2.3 void Tetris::Board::drawBackground (Canvas * *c*) const [protected]

6.2.2.4 void Tetris::Board::drawCurrShape (Canvas * *c*) const [protected]

6.2.2.5 void Tetris::Board::drawFragments (Canvas * *c*) const [protected]

6.2.2.6 void Tetris::Board::drawShape (int *i*, int *j*, Canvas * *cnvs*) const

6.2.2.7 void Tetris::Board::drawUnitSquare (Fl_Color *clr1*, int *i*, int *j*, bool *dotP*, Fl_Color *clr2*, Canvas * *cnvs*) const

6.2.2.8 vector< TCode > Tetris::Board::emptyBoard ( ) const [protected]

6.2.2.9 unsigned int Tetris::Board::nFromIJ (int *i*, int *j*) const

6.2.2.10 void Tetris::Board::placeShape (Shape *s*, int *i*, int *j*) [protected]

6.2.2.11 void Tetris::Board::randomizeFragments (double *f*)

6.2.2.12 void Tetris::Board::randomizeRow (unsigned int *i*) [protected]

6.2.2.13 bool Tetris::Board::resetCurrPiece ( )

6.2.2.14 void Tetris::Board::rotateDown ( )

6.2.2.15 unsigned int Tetris::Board::stepGame ( )

6.2.2.16 bool Tetris::Board::testSDrop ( )

6.2.2.17 bool Tetris::Board::testShape (Shape *s*, int *i*, int *j*) const [protected]

6.2.2.18 bool Tetris::Board::tryHDrop ( )

6.2.2.19 bool Tetris::Board::tryLMove ( )

6.2.2.20 bool Tetris::Board::tryLRot ( )

6.2.2.21 bool Tetris::Board::tryRMove ( )

6.2.2.22 bool Tetris::Board::tryRRot ( )

6.2.2.23 bool Tetris::Board::trySDrop ( )

6.2.2.24 void Tetris::Board::update (Canvas * *c*) const [virtual]

Reimplemented from KGraph::Picture.

6.2.3 Member Data Documentation

6.2.3.1 unsigned int Tetris::Board::clms = 0

6.2.3.2 int Tetris::Board::currI = 0

6.2.3.3 int Tetris::Board::currJ = 0

6.2.3.4 Shape Tetris::Board::currShape = Shape()

6.2.3.5 Shape Tetris::Board::nextShape = Shape()

6.2.3.6 unsigned int Tetris::Board::rows = 0

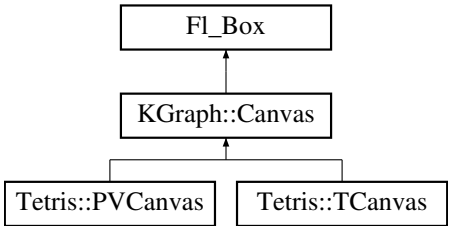
The documentation for this class was generated from the following files:

- board.h
- board.cpp

6.3 KGraph::Canvas Class Reference

`#include <kgraph.h>`

Inheritance diagram for KGraph::Canvas:



Public Member Functions

- Canvas (int x, int y, int w, int h, const char *l=0)

  Abstract base class.
- virtual ~Canvas ()
- void end ()
- void updateMaps ()
- void clearMaps ()
- virtual int handle (int ev)
- virtual void onMove (int x, int y)
- virtual void onDrag (int x, int y)
- virtual void onPush (int x, int y, int b)
- virtual void onRelease (int x, int y, int b)
- virtual void onKeyDown (int x, int y, int k)

Public Attributes

- Picture * pict = nullptr
- CoordMap * xMap = nullptr
- CoordMap * yMap = nullptr

6.3.1 Constructor & Destructor Documentation

6.3.1.1 KGraph::Canvas::Canvas (int *x*, int *y*, int *w*, int *h*, const char * *l* = 0)

Abstract base class.

6.3.1.2 KGraph::Canvas::~Canvas ( ) [virtual]

6.3.2 Member Function Documentation

6.3.2.1 void KGraph::Canvas::clearMaps ( )

6.3.2.2 void KGraph::Canvas::end ( )

6.3.2.3 int KGraph::Canvas::handle (int *ev*) [virtual]

6.3.2.4 void KGraph::Canvas::onDrag (int *x*, int *y*) [virtual]

Reimplemented in Tetris::PVCanvas, and Tetris::TCanvas.

6.3.2.5 void KGraph::Canvas::onKeyDown (int *x*, int *y*, int *k*) [virtual]

Reimplemented in Tetris::PVCanvas, and Tetris::TCanvas.

6.3.2.6 void KGraph::Canvas::onMove (int *x*, int *y*) [virtual]

Reimplemented in Tetris::PVCanvas, and Tetris::TCanvas.

6.3.2.7 void KGraph::Canvas::onPush (int *x*, int *y*, int *b*) [virtual]

Reimplemented in Tetris::PVCanvas, and Tetris::TCanvas.

6.3.2.8 void KGraph::Canvas::onRelease (int *x*, int *y*, int *b*) [virtual]

Reimplemented in Tetris::PVCanvas, and Tetris::TCanvas.

6.3.2.9 void KGraph::Canvas::updateMaps ( )

6.3.3 Member Data Documentation

6.3.3.1 Picture* KGraph::Canvas::pict = nullptr

6.3.3.2 CoordMap* KGraph::Canvas::xMap = nullptr

6.3.3.3 CoordMap* KGraph::Canvas::yMap = nullptr

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

- kgraph.h
- kgraph.cpp

6.4 Tetris::ControlState Class Reference

```
#include <tmain.h>
```

Public Member Functions

- ControlState ()
- ControlState (unsigned int bv, unsigned int pv, unsigned int gv, unsigned int rv)

Public Attributes

- unsigned int bg = 1
- unsigned int pc = 2
- unsigned int gt = 1
- unsigned int rt = 0

6.4.1 Constructor & Destructor Documentation

6.4.1.1 Tetris::ControlState::ControlState ( ) [inline]

6.4.1.2 Tetris::ControlState::ControlState (unsigned int *bv*, unsigned int *pv*, unsigned int *gv*, unsigned int *rv*) [inline]

6.4.2 Member Data Documentation

6.4.2.1 unsigned int Tetris::ControlState::bg = 1

6.4.2.2 unsigned int Tetris::ControlState::gt = 1

6.4.2.3 unsigned int Tetris::ControlState::pc = 2

6.4.2.4 unsigned int Tetris::ControlState::rt = 0

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

- tmain.h

6.5 KGraph::CoordMap Class Reference

`#include <kgraph.h>`

Public Member Functions

- CoordMap (int s1, double d1, int s2, double d2)
- virtual ~CoordMap ()
- int d2s (double d)
- double s2d (int s)

Protected Attributes

- double as = 0
- double bs = 0
- double ad = 0
- double bd = 0

6.5.1 Constructor & Destructor Documentation

6.5.1.1 KGraph::CoordMap::CoordMap (int *s1,* double *d1,* int *s2,* double *d2*)

6.5.1.2 KGraph::CoordMap::~CoordMap ( ) [virtual]

6.5.2 Member Function Documentation

6.5.2.1 int KGraph::CoordMap::d2s (double *d*)

6.5.2.2 double KGraph::CoordMap::s2d (int *s*)

6.5.3 Member Data Documentation

6.5.3.1 double KGraph::CoordMap::ad = 0 [protected]

6.5.3.2 double KGraph::CoordMap::as = 0 [protected]

6.5.3.3 double KGraph::CoordMap::bd = 0 [protected]

6.5.3.4 double KGraph::CoordMap::bs = 0 [protected]

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

- kgraph.h
- kgraph.cpp

6.6 KBase::GAOpt< GAP > Class Template Reference

`#include <gaopt.h>`

Public Member Functions

- GAOpt (unsigned int s)
- virtual ~GAOpt ()
- void init (vector< GAP * > ipop)
- void fill (PRNG *rng)
- void run (PRNG *rng, double c, double m, unsigned int maxI, double sTh, unsigned int maxS, ReportingLevel srl, unsigned int &iter, unsigned int &sIter)
- tuple< double, GAP * > getNth (unsigned int n)
- void show ()
- void sortPop ()

Public Attributes

- function< tuple< GAP *, GAP * >const GAP *g1, const GAP *g2, PRNG *rng)> cross = nullptr
- function< GAP *(const GAP *g1, PRNG *rng)> mutate = nullptr
- function< double(const GAP *g1)> eval = nullptr
- function< void(const GAP *)> showGene = nullptr
- function< GAP *(PRNG *rng)> makeGene = nullptr
- function< bool(const GAP *g1, const GAP *g2)> equiv = nullptr

Protected Member Functions

- void step ()
- void mutatePop ()
- void crossPop ()
- void dropDups ()
- void selectPop ()
- GAP * mutateOne (const GAP *g1, PRNG *rng)
- tuple< GAP *, GAP * > crossPair (const GAP *g1, const GAP *g2, PRNG *rng)
- void cyclicApply (function< void(unsigned int i)> fn, double f)

Protected Attributes

- vector< tuple< double, GAP * > > gpool = {}
- unsigned int pSize = 0
- double cFrac = 1.0
- double mFrac = 0.5
- PRNG * rng = nullptr

6.6.1 Constructor & Destructor Documentation

6.6.1.1 template<class GAP > KBase::GAOpt< GAP >::GAOpt (unsigned int *s*) [explicit]

6.6.1.2 template<class GAP > KBase::GAOpt< GAP >::~GAOpt ( ) [virtual]

6.6.2 Member Function Documentation

6.6.2.1 template<class GAP > tuple<GAP*, GAP*> KBase::GAOpt< GAP >::crossPair (const GAP * *g1,* const GAP * *g2,* PRNG * *rng*) [protected]

6.6.2.2 template<class GAP > void KBase::GAOpt< GAP >::crossPop ( ) [protected]

6.6.2.3 template<class GAP > void KBase::GAOpt< GAP >::cyclicApply (function< void(unsigned int i)> *fn,* double *f*) [protected]

6.6.2.4 template<class GAP > void KBase::GAOpt< GAP >::dropDups ( ) [protected]

6.6.2.5 template<class GAP > void KBase::GAOpt< GAP >::fill (PRNG * *rng*)

6.6.2.6 template<class GAP > tuple< double, GAP * > KBase::GAOpt< GAP >::getNth (unsigned int *n*)

6.6.2.7 template<class GAP > void KBase::GAOpt< GAP >::init (vector< GAP * > *ipop*)

6.6.2.8 template<class GAP > GAP* KBase::GAOpt< GAP >::mutateOne (const GAP * *g1,* PRNG * *rng*) [protected]

6.6.2.9 template<class GAP > void KBase::GAOpt< GAP >::mutatePop ( ) [protected]

6.6.2.10 template<class GAP > void KBase::GAOpt< GAP >::run (PRNG * *rng,* double *c,* double *m,* unsigned int *maxI,* double *sTh,* unsigned int *maxS,* ReportingLevel *srl,* unsigned int & *iter,* unsigned int & *sIter*)

6.6.2.11 template<class GAP > void KBase::GAOpt< GAP >::selectPop ( ) [protected]

6.6.2.12 template<class GAP > void KBase::GAOpt< GAP >::show ( )

6.6.2.13 template<class GAP > void KBase::GAOpt< GAP >::sortPop ( )

6.6.2.14 template<class GAP > void KBase::GAOpt< GAP >::step ( ) [protected]

6.6.3 Member Data Documentation

6.6.3.1 template<class GAP > double KBase::GAOpt< GAP >::cFrac = 1.0 [protected]

6.6.3.2 template<class GAP > function<tuple<GAP*, GAP*>const GAP* g1, const GAP* g2, PRNG* rng)> KBase::GAOpt< GAP >::cross = nullptr

6.6.3.3 template<class GAP > function<bool(const GAP* g1, const GAP* g2)> KBase::GAOpt< GAP >::equiv = nullptr

6.6.3.4 template<class GAP > function<double(const GAP* g1)> KBase::GAOpt< GAP >::eval = nullptr

6.6.3.5 template<class GAP > vector< tuple<double, GAP* > > KBase::GAOpt< GAP >::gpool = {} [protected]

6.6.3.6 template<class GAP > function<GAP* (PRNG* rng)> KBase::GAOpt< GAP >::makeGene = nullptr

6.6.3.7 template<class GAP > double KBase::GAOpt< GAP >::mFrac = 0.5 [protected]

6.6.3.8 template<class GAP > function<GAP* (const GAP* g1, PRNG* rng)> KBase::GAOpt< GAP >::mutate = nullptr

6.6.3.9 template<class GAP > unsigned int KBase::GAOpt< GAP >::pSize = 0 [protected]

6.6.3.10 template<class GAP > PRNG* KBase::GAOpt< GAP >::rng = nullptr [protected]

6.6.3.11 template<class GAP > function<void(const GAP*)> KBase::GAOpt< GAP >::showGene = nullptr

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

- gaopt.h

6.7 KBase::GHCSearch< HCP > Class Template Reference

`#include <hcsearch.h>`

Public Member Functions

- GHCSearch ()
- virtual ~GHCSearch ()
- tuple< double, HCP, unsigned int, unsigned int > run (HCP p0, ReportingLevel srl, unsigned int iMax, unsigned int sMax, double sTol)

Public Attributes

- function< double(const HCP)> eval = nullptr
- function< vector< HCP >const HCP)> nghbrs = nullptr
- function< void(const HCP)> show = nullptr

6.7.1 Constructor & Destructor Documentation

6.7.1.1 template<class HCP > KBase::GHCSearch< HCP >::GHCSearch ( )

6.7.1.2 template<class HCP > KBase::GHCSearch< HCP >::~GHCSearch ( ) [virtual]

6.7.2 Member Function Documentation

6.7.2.1 template<class HCP > tuple< double, HCP, unsigned int, unsigned int > KBase::GHCSearch< HCP >::run (HCP *p0,* ReportingLevel *srl,* unsigned int *iMax,* unsigned int *sMax,* double *sTol*)

6.7.3 Member Data Documentation

6.7.3.1 template<class HCP> function<double(const HCP)> KBase::GHCSearch< HCP >::eval = nullptr

6.7.3.2 template<class HCP> function<vector<HCP>const HCP)> KBase::GHCSearch< HCP >::nghbrs = nullptr

6.7.3.3 template<class HCP> function<void(const HCP)> KBase::GHCSearch< HCP >::show = nullptr

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

- hcsearch.h

6.8 KBase::KException Class Reference

`#include <kutils.h>`

Public Member Functions

- KException (string m)
- virtual ~KException ()

Public Attributes

- string msg =""

6.8.1 Constructor & Destructor Documentation

6.8.1.1 KBase::KException::KException (string *m*) [explicit]

6.8.1.2 KBase::KException::~KException ( ) [virtual]

6.8.2 Member Data Documentation

6.8.2.1 string KBase::KException::msg =""

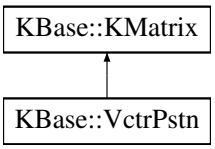
The documentation for this class was generated from the following files:

- kutils.h
- kutils.cpp

6.9 KBase::KMatrix Class Reference

`#include <kmatrix.h>`

Inheritance diagram for KBase::KMatrix:



Public Member Functions

- KMatrix ()
- KMatrix (unsigned int nr, unsigned int nc)
- double operator() (unsigned int i, unsigned int j) const
- double & operator() (unsigned int i, unsigned int j)
- void mPrintf (string) const
- unsigned int numR () const
- unsigned int numC () const
- vector< double >::iterator begin ()
- vector< double >::iterator end ()

- vector< double >::const_iterator cbegin ()
- vector< double >::const_iterator cend ()
- vector< double >::const_iterator begin () const
- vector< double >::const_iterator end () const
- virtual ~KMatrix ()

Static Public Member Functions

- static KMatrix uniform (PRNG *rng, unsigned int nr, unsigned int nc, double a, double b)
- static KMatrix map (function< double(unsigned int i, unsigned int j)> f, unsigned int nr, unsigned int nc)
- static void mapV (function< void(unsigned int i, unsigned int j)> f, unsigned int nr, unsigned int nc)
- static KMatrix arrayInit (const double mv[], const unsigned int &rows, const unsigned int &clms)

Friends

- KMatrix inv (const KMatrix &m)

6.9.1 Constructor & Destructor Documentation

6.9.1.1 KBase::KMatrix::KMatrix ( )

6.9.1.2 KBase::KMatrix::KMatrix (unsigned int *nr,* unsigned int *nc*)

6.9.1.3 KBase::KMatrix::~KMatrix ( ) [virtual]

6.9.2 Member Function Documentation

6.9.2.1 KMatrix KBase::KMatrix::arrayInit (const double *mv[],* const unsigned int & *rows,* const unsigned int & *clms*) [static]

6.9.2.2 vector<double>::iterator KBase::KMatrix::begin ( ) [inline]

6.9.2.3 vector<double>::const_iterator KBase::KMatrix::begin ( ) const [inline]

6.9.2.4 vector<double>::const_iterator KBase::KMatrix::cbegin ( ) [inline]

6.9.2.5 vector<double>::const_iterator KBase::KMatrix::cend ( ) [inline]

6.9.2.6 vector<double>::iterator KBase::KMatrix::end ( ) [inline]

6.9.2.7 vector<double>::const_iterator KBase::KMatrix::end ( ) const [inline]

6.9.2.8 KMatrix KBase::KMatrix::map (function< double(unsigned int i, unsigned int j)> *f,* unsigned int *nr,* unsigned int *nc*) [static]

6.9.2.9 void KBase::KMatrix::mapV (function< void(unsigned int i, unsigned int j)> *f,* unsigned int *nr,* unsigned int *nc*) [static]

6.9.2.10 void KBase::KMatrix::mPrintf (string *fs*) const

6.9.2.11 unsigned int KBase::KMatrix::numC ( ) const

6.9.2.12 unsigned int KBase::KMatrix::numR ( ) const

6.9.2.13 double KBase::KMatrix::operator() (unsigned int *i,* unsigned int *j*) const

6.9.2.14 double & KBase::KMatrix::operator() (unsigned int *i,* unsigned int *j*)

6.9.2.15 KMatrix KBase::KMatrix::uniform (PRNG * *rng,* unsigned int *nr,* unsigned int *nc,* double *a,* double *b*) [static]

6.9.3 Friends And Related Function Documentation

6.9.3.1 KMatrix inv (const KMatrix & *m*) [friend]

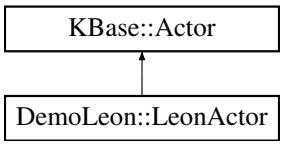
The documentation for this class was generated from the following files:

- kmatrix.h
- kmatrix.cpp

6.10 DemoLeon::LeonActor Class Reference

`#include <demoleon.h>`

Inheritance diagram for DemoLeon::LeonActor:



Public Member Functions

- LeonActor (string n, string d, LeonModel *em, unsigned int id)
- ~LeonActor ()
- double vote (unsigned int p1, unsigned int p2, const State *st) const
- virtual double vote (const Position *ap1, const Position *ap2) const
- double posUtil (const Position *ap1) const
- void randomize (PRNG *rng)
- void setShareUtilScale (const KMatrix &runs)
- double shareToUtil (double gdpShare) const

Public Attributes

- const LeonModel * eMod = nullptr
- unsigned int idNum = 0
- KMatrix vCap = KMatrix()
- VotingRule vr = VotingRule::Proportional
- double minS = 0
- double refS = 0.5
- double refU = 0.5
- double maxS = 1

Additional Inherited Members

6.10.1 Constructor & Destructor Documentation

6.10.1.1 DemoLeon::LeonActor::LeonActor (string *n*, string *d*, LeonModel * *em*, unsigned int *id*)

6.10.1.2 DemoLeon::LeonActor::~LeonActor ( )

6.10.2 Member Function Documentation

6.10.2.1 double DemoLeon::LeonActor::posUtil (const Position * *ap1*) const

6.10.2.2 void DemoLeon::LeonActor::randomize (PRNG * *rng*)

6.10.2.3 void DemoLeon::LeonActor::setShareUtilScale (const KMatrix & *runs*)

6.10.2.4 double DemoLeon::LeonActor::shareToUtil (double *gdpShare*) const

6.10.2.5 double DemoLeon::LeonActor::vote (unsigned int *p1*, unsigned int *p2*, const State * *st*) const [virtual]

Implements KBase::Actor.

6.10.2.6 double DemoLeon::LeonActor::vote (const Position * *ap1*, const Position * *ap2*) const [virtual]

6.10.3 Member Data Documentation

6.10.3.1 const LeonModel* DemoLeon::LeonActor::eMod = nullptr

6.10.3.2 unsigned int DemoLeon::LeonActor::idNum = 0

6.10.3.3 double DemoLeon::LeonActor::maxS = 1

6.10.3.4 double DemoLeon::LeonActor::minS = 0

6.10.3.5 double DemoLeon::LeonActor::refS = 0.5

6.10.3.6 double DemoLeon::LeonActor::refU = 0.5

6.10.3.7 KMatrix DemoLeon::LeonActor::vCap = KMatrix()

6.10.3.8 VotingRule DemoLeon::LeonActor::vr = VotingRule::Proportional

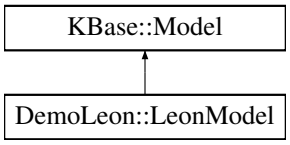
The documentation for this class was generated from the following files:

- demoleon.h
- demoleon.cpp

6.11 DemoLeon::LeonModel Class Reference

`#include <demoleon.h>`

Inheritance diagram for DemoLeon::LeonModel:



Public Member Functions

- LeonModel (PRNG *r, string d="")
- virtual ~LeonModel ()
- tuple< KMatrix, KMatrix, KMatrix, KMatrix > makeBaseYear (unsigned int numF, unsigned int numCG, unsigned int numS, PRNG *rng)
- void makeIOModel (const KMatrix &trns, const KMatrix &rev, const KMatrix &xprt, const KMatrix &cons, PRNG *rng)
- KMatrix xprtDemand (const KBase::KMatrix &tau) const
- KMatrix randomFTax (PRNG *rng)
- KMatrix makeFTax (const KBase::KMatrix &tax) const
- double infsDegree (const KMatrix &tax) const
- KMatrix vaShares (const KMatrix &tax, bool normalizeSharesP) const
- KMatrix monteCarloShares (unsigned int nRuns, KBase::PRNG *rng)

Static Public Member Functions

- static double stateDist (const LeonState *s1, const LeonState *s2)

Public Attributes

- double posTol = 1E-5

  how close together positions must be to be considered equivalent

Protected Attributes

- unsigned int L = 0
- unsigned int M = 0
- unsigned int N = 0
- double maxSub = 0.5
- double maxTax = 0.5
- KMatrix x0 = KMatrix()
- KMatrix eps = KMatrix()
- KMatrix aL = KMatrix()
- KMatrix bL = KMatrix()
- KMatrix rho = KMatrix()
- KMatrix vas = KMatrix()

Friends

- class LeonActor

Additional Inherited Members

6.11.1 Constructor & Destructor Documentation

6.11.1.1 DemoLeon::LeonModel::LeonModel (PRNG * *r,* string *d* = "") [explicit]

6.11.1.2 DemoLeon::LeonModel::~LeonModel ( ) [virtual]

6.11.2 Member Function Documentation

6.11.2.1 double DemoLeon::LeonModel::infsDegree (const KMatrix & *tax*) const

6.11.2.2 tuple< KMatrix, KMatrix, KMatrix, KMatrix > DemoLeon::LeonModel::makeBaseYear (unsigned int *numF,* unsigned int *numCG,* unsigned int *numS,* PRNG * *rng*)

6.11.2.3 KMatrix DemoLeon::LeonModel::makeFTax (const KBase::KMatrix & *tax*) const

6.11.2.4 void DemoLeon::LeonModel::makeIOModel (const KMatrix & *trns,* const KMatrix & *rev,* const KMatrix & *xprt,* const KMatrix & *cons,* PRNG * *rng*)

6.11.2.5 KMatrix DemoLeon::LeonModel::monteCarloShares (unsigned int *nRuns,* KBase::PRNG * *rng*)

6.11.2.6 KMatrix DemoLeon::LeonModel::randomFTax (PRNG * *rng*)

6.11.2.7 double DemoLeon::LeonModel::stateDist (const LeonState * *s1,* const LeonState * *s2*) [static]

6.11.2.8 KMatrix DemoLeon::LeonModel::vaShares (const KMatrix & *tax,* bool *normalizeSharesP*) const

6.11.2.9 KMatrix DemoLeon::LeonModel::xprtDemand (const KBase::KMatrix & *tau*) const

6.11.3 Friends And Related Function Documentation

6.11.3.1 friend class LeonActor [friend]

6.11.4 Member Data Documentation

6.11.4.1 KMatrix DemoLeon::LeonModel::aL = KMatrix() [protected]

6.11.4.2 KMatrix DemoLeon::LeonModel::bL = KMatrix() [protected]

6.11.4.3 KMatrix DemoLeon::LeonModel::eps = KMatrix() [protected]

6.11.4.4 unsigned int DemoLeon::LeonModel::L = 0 [protected]

6.11.4.5 unsigned int DemoLeon::LeonModel::M = 0 [protected]

6.11.4.6 double DemoLeon::LeonModel::maxSub = 0.5 [protected]

6.11.4.7 double DemoLeon::LeonModel::maxTax = 0.5 [protected]

6.11.4.8 unsigned int DemoLeon::LeonModel::N = 0 [protected]

6.11.4.9 double DemoLeon::LeonModel::posTol = 1E-5

how close together positions must be to be considered equivalent

6.11.4.10 KMatrix DemoLeon::LeonModel::rho = KMatrix() [protected]

6.11.4.11 KMatrix DemoLeon::LeonModel::vas = KMatrix() [protected]

6.11.4.12 KMatrix DemoLeon::LeonModel::x0 = KMatrix() [protected]

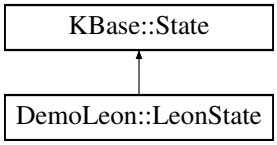
The documentation for this class was generated from the following files:

- demoleon.h
- demoleon.cpp

6.12 DemoLeon::LeonState Class Reference

`#include <demoleon.h>`

Inheritance diagram for DemoLeon::LeonState:



Public Member Functions

- LeonState (LeonModel *em)
- ~LeonState ()
- virtual tuple< KMatrix, vector< unsigned int > > pDist (int persp) const
- virtual void setAUtil (ReportingLevel rl)
- LeonState * stepSUSN ()

Public Attributes

- const LeonModel * eMod = nullptr

Protected Member Functions

- LeonState * doSUSN (ReportingLevel rl) const
- virtual bool equivNdx (unsigned int i, unsigned int j) const

6.12.1 Constructor & Destructor Documentation

6.12.1.1 DemoLeon::LeonState::LeonState (LeonModel * *em*) [explicit]

6.12.1.2 DemoLeon::LeonState::~LeonState ( )

6.12.2 Member Function Documentation

6.12.2.1 LeonState * DemoLeon::LeonState::doSUSN (ReportingLevel *rl*) const [protected]

6.12.2.2 bool DemoLeon::LeonState::equivNdx (unsigned int *i*, unsigned int *j*) const [protected], [virtual]

Compare two actual positions in the current state

Implements KBase::State.

6.12.2.3 tuple< KMatrix, vector< unsigned int > > DemoLeon::LeonState::pDist (int *persp*) const [virtual]

Implements KBase::State.

6.12.2.4 void DemoLeon::LeonState::setAUtil (ReportingLevel *rl*) [virtual]

6.12.2.5 LeonState * DemoLeon::LeonState::stepSUSN ( )

6.12.3 Member Data Documentation

6.12.3.1 const LeonModel* DemoLeon::LeonState::eMod = nullptr

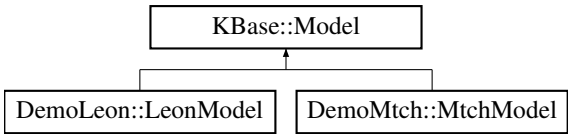
The documentation for this class was generated from the following files:

- demoleon.h
- demoleon.cpp

6.13 KBase::Model Class Reference

`#include <kmodel.h>`

Inheritance diagram for KBase::Model:



Public Types

- enum VPModel { VPModel::Linear, VPModel::Square, VPModel::Quartic, VPModel::Binary }

Public Member Functions

- Model (PRNG *r, string d)
- virtual ~Model ()
- void run ()
- virtual unsigned int addActor (Actor *a)
- int actrNdx (const Actor *a) const
- int addState (State *s)
- void sqlAUtil (unsigned int t)

Static Public Member Functions

- static double nProd (double x, double y)
- static double vote (VotingRule vr, double wi, double uij, double uik)
- static KMatrix coalitions (function< double(unsigned int ak, unsigned int pi, unsigned int pj)> vfn, unsigned int numAct, unsigned int numOpt)
- static string VPMName (VPModel vpm)
- static KMatrix vProb (VPModel vpm, const KMatrix &c)
- static tuple< double, double > vProb (VPModel vpm, const double s1, const double s2)
- static KMatrix vProb (VotingRule vr, VPModel vpm, const KMatrix &w, const KMatrix &u)
- static KMatrix probCE (const KMatrix &pv)
- static KMatrix markovPCE (const KMatrix &pv)
- static KMatrix condPCE (const KMatrix &pv)
- static KMatrix scalarPCE (unsigned int numAct, unsigned int numOpt, const KMatrix &w, const KMatrix &u, VotingRule vr, VPModel vpm, ReportingLevel rl)
- static void demoSQLite ()

Public Attributes

- function< bool(unsigned int iter, const State *s)> stop = nullptr
- vector< Actor * > actrs = {}
- unsigned int numAct = 0
- PRNG * rng = nullptr
- vector< State * > history = {}

Static Protected Member Functions

- static string createTableSQL (unsigned int tn)

Protected Attributes

- sqlite3 * smpDB = nullptr
- string scenName = "Scen"

6.13.1 Member Enumeration Documentation

6.13.1.1 enum KBase::Model::VPModel [strong]

Enumerator

Linear

Square

Quartic

Binary

6.13.2 Constructor & Destructor Documentation

6.13.2.1 KBase::Model::Model (PRNG * *r*, string *d*) [explicit]

6.13.2.2 KBase::Model::~Model ( ) [virtual]

6.13.3 Member Function Documentation

6.13.3.1 int KBase::Model::actrNdx (const Actor * *a*) const

6.13.3.2 unsigned int KBase::Model::addActor (Actor * *a*) [virtual]

6.13.3.3 int KBase::Model::addState (State * *s*)

6.13.3.4 KMatrix KBase::Model::coalitions (function< double(unsigned int ak, unsigned int pi, unsigned int pj)> *vfn,* unsigned int *numAct,* unsigned int *numOpt*) [static]

6.13.3.5 KMatrix KBase::Model::condPCE (const KMatrix & *pv*) [static]

6.13.3.6 string KBase::Model::createTableSQL (unsigned int *tn*) [static], [protected]

6.13.3.7 void KBase::Model::demoSQLite ( ) [static]

6.13.3.8 KMatrix KBase::Model::markovPCE (const KMatrix & *pv*) [static]

6.13.3.9 double KBase::Model::nProd (double *x,* double *y*) [static]

6.13.3.10 KMatrix KBase::Model::probCE (const KMatrix & *pv*) [static]

6.13.3.11 void KBase::Model::run ( )

6.13.3.12 KMatrix KBase::Model::scalarPCE (unsigned int *numAct,* unsigned int *numOpt,* const KMatrix & *w,* const KMatrix & *u,* VotingRule *vr,* VPModel *vpm,* ReportingLevel *rl*) [static]

6.13.3.13 void KBase::Model::sqlAUtil (unsigned int *t*)

6.13.3.14 double KBase::Model::vote (VotingRule *vr,* double *wi,* double *uij,* double *uik*) [static]

6.13.3.15 string KBase::Model::VPMName (VPModel *vpm*) [static]

6.13.3.16 KMatrix KBase::Model::vProb (VPModel *vpm,* const KMatrix & *c*) [static]

6.13.3.17 tuple< double, double > KBase::Model::vProb (VPModel *vpm,* const double *s1,* const double *s2*) [static]

6.13.3.18 KMatrix KBase::Model::vProb (VotingRule *vr,* VPModel *vpm,* const KMatrix & *w,* const KMatrix & *u*) [static]

6.13.4 Member Data Documentation

6.13.4.1 vector<Actor*> KBase::Model::actrs = {}

6.13.4.2 vector<State*> KBase::Model::history = {}

6.13.4.3 unsigned int KBase::Model::numAct = 0

6.13.4.4 PRNG* KBase::Model::rng = nullptr

6.13.4.5 string KBase::Model::scenName = "Scen" [protected]

6.13.4.6 sqlite3* KBase::Model::smpDB = nullptr [protected]

6.13.4.7 function<bool(unsigned int iter, const State* s)> KBase::Model::stop = nullptr

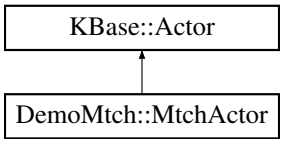
The documentation for this class was generated from the following files:

- kmodel.h
- kmodel.cpp

6.14 DemoMtch::MtchActor Class Reference

`#include <demomtch.h>`

Inheritance diagram for DemoMtch::MtchActor:



Public Types

- enum PropModel { PropModel::ExpUtil, PropModel::Probability, PropModel::AgreeUtil }

Public Member Functions

- MtchActor (string n, string d)
- ~MtchActor ()
- double vote (unsigned int p1, unsigned int p2, const State *st) const
- virtual double vote (const Position *ap1, const Position *ap2) const
- double posUtil (const Position *ap1) const
- void randomize (PRNG *rng, double minCap, double maxCap, unsigned int id, unsigned int numI)
- tuple< double, MtchPstn > maxProbEUPstn (PropModel pm, const MtchState *mst) const

Static Public Member Functions

- static MtchPstn * rPos (unsigned int numI, unsigned int numA, PRNG *rng)
- static MtchActor * rAct (unsigned int numI, double minCap, double maxCap, PRNG *rng, unsigned int i)

Public Attributes

- unsigned int idNum = 0
- VotingRule vr = VotingRule::Proportional
- PropModel pMod = PropModel::ExpUtil
- double sCap = 0
- vector< double > vals = {}

6.14.1 Member Enumeration Documentation

6.14.1.1 enum DemoMtch::MtchActor::PropModel [strong]

Enumerator

ExpUtil

Probability

AgreeUtil

6.14.2 Constructor & Destructor Documentation

6.14.2.1 DemoMtch::MtchActor::MtchActor (string *n*, string *d*)

6.14.2.2 DemoMtch::MtchActor::~MtchActor ( )

6.14.3 Member Function Documentation

6.14.3.1 tuple< double, MtchPstn > DemoMtch::MtchActor::maxProbEUPstn (PropModel *pm*, const MtchState * *mst*) const

6.14.3.2 double DemoMtch::MtchActor::posUtil (const Position * *ap1*) const

6.14.3.3 MtchActor * DemoMtch::MtchActor::rAct (unsigned int *numI*, double *minCap*, double *maxCap*, PRNG * *rng*, unsigned int *i*) [static]

6.14.3.4 void DemoMtch::MtchActor::randomize (PRNG * *rng*, double *minCap*, double *maxCap*, unsigned int *id*, unsigned int *numI*)

6.14.3.5 MtchPstn * DemoMtch::MtchActor::rPos (unsigned int *numI*, unsigned int *numA*, PRNG * *rng*) [static]

6.14.3.6 double DemoMtch::MtchActor::vote (unsigned int *p1*, unsigned int *p2*, const State * *st*) const [virtual]

Implements KBase::Actor.

6.14.3.7 double DemoMtch::MtchActor::vote (const Position * *ap1*, const Position * *ap2*) const [virtual]

6.14.4 Member Data Documentation

6.14.4.1 unsigned int DemoMtch::MtchActor::idNum = 0

6.14.4.2 PropModel DemoMtch::MtchActor::pMod = PropModel::ExpUtil

6.14.4.3 double DemoMtch::MtchActor::sCap = 0

6.14.4.4 vector<double> DemoMtch::MtchActor::vals = {}

6.14.4.5 VotingRule DemoMtch::MtchActor::vr = VotingRule::Proportional

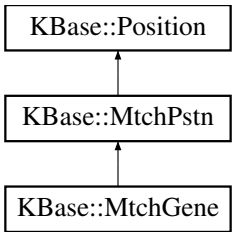
The documentation for this class was generated from the following files:

- demomtch.h
- demomtch.cpp

6.15 KBase::MtchGene Class Reference

`#include <kmodel.h>`

Inheritance diagram for KBase::MtchGene:



Public Member Functions

- MtchGene ()
- ~MtchGene ()
- void randomize (PRNG *rng)
- MtchGene * mutate (PRNG *rng) const
- tuple< MtchGene *, MtchGene * > cross (const MtchGene *g2, PRNG *rng) const
- void show () const
- bool equiv (const MtchGene *g2) const
- void setState (vector< Actor * > as, vector< MtchPstn * > ps)

Protected Member Functions

- void copySelf (MtchGene *) const

Protected Attributes

- vector< Actor * > actrs = {}
- vector< MtchPstn * > pstns = {}

Additional Inherited Members

6.15.1 Constructor & Destructor Documentation

6.15.1.1 KBase::MtchGene::MtchGene ( )

6.15.1.2 KBase::MtchGene::~MtchGene ( )

6.15.2 Member Function Documentation

6.15.2.1 void KBase::MtchGene::copySelf (MtchGene * *mg2*) const [protected]

6.15.2.2 tuple< MtchGene *, MtchGene * > KBase::MtchGene::cross (const MtchGene * *g2*, PRNG * *rng*) const

6.15.2.3 bool KBase::MtchGene::equiv (const MtchGene * *g2*) const

6.15.2.4 MtchGene * KBase::MtchGene::mutate (PRNG * *rng*) const

6.15.2.5 void KBase::MtchGene::randomize (PRNG * *rng*)

6.15.2.6 void KBase::MtchGene::setState (vector< Actor * > *as*, vector< MtchPstn * > *ps*)

6.15.2.7 void KBase::MtchGene::show ( ) const

6.15.3 Member Data Documentation

6.15.3.1 vector<Actor*> KBase::MtchGene::actrs = {} [protected]

6.15.3.2 vector<MtchPstn*> KBase::MtchGene::pstns = {} [protected]

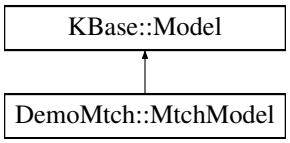
The documentation for this class was generated from the following files:

- kmodel.h
- kposition.cpp

6.16 DemoMtch::MtchModel Class Reference

`#include <demomtch.h>`

Inheritance diagram for DemoMtch::MtchModel:



Public Member Functions

- MtchModel (PRNG *rng, string d="")
- virtual ~MtchModel ()

Static Public Member Functions

- static MtchModel * randomMS (unsigned int numA, unsigned int numI, VotingRule vr, MtchActor::PropModel pMod, PRNG *rng)

Public Attributes

- unsigned int numItm = 0
- unsigned int numCat = 0

Additional Inherited Members

6.16.1 Constructor & Destructor Documentation

6.16.1.1 DemoMtch::MtchModel::MtchModel (PRNG * *rng*, string *d* = " ") [explicit]

6.16.1.2 DemoMtch::MtchModel::~MtchModel ( ) [virtual]

6.16.2 Member Function Documentation

6.16.2.1 MtchModel * DemoMtch::MtchModel::randomMS (unsigned int *numA*, unsigned int *numI*, VotingRule *vr*, MtchActor::PropModel *pMod*, PRNG * *rng*) [static]

6.16.3 Member Data Documentation

6.16.3.1 unsigned int DemoMtch::MtchModel::numCat = 0

6.16.3.2 unsigned int DemoMtch::MtchModel::numItm = 0

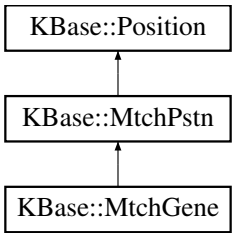
The documentation for this class was generated from the following files:

- demomtch.h
- demomtch.cpp

6.17 KBase::MtchPstn Class Reference

`#include <kmodel.h>`

Inheritance diagram for KBase::MtchPstn:



Public Member Functions

- MtchPstn ()
- virtual ~MtchPstn ()
- virtual vector< MtchPstn > neighbors (unsigned int nVar) const

Public Attributes

- unsigned int numItm = 0
- unsigned int numCat = 0
- vector< unsigned int > match = {}

6.17.1 Constructor & Destructor Documentation

6.17.1.1 KBase::MtchPstn::MtchPstn ( )

6.17.1.2 KBase::MtchPstn::~MtchPstn ( ) [virtual]

6.17.2 Member Function Documentation

6.17.2.1 vector< MtchPstn > KBase::MtchPstn::neighbors (unsigned int *nVar*) const [virtual]

6.17.3 Member Data Documentation

6.17.3.1 vector<unsigned int> KBase::MtchPstn::match = {}

6.17.3.2 unsigned int KBase::MtchPstn::numCat = 0

6.17.3.3 unsigned int KBase::MtchPstn::numItm = 0

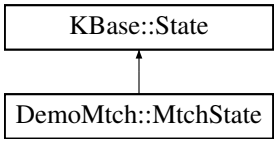
The documentation for this class was generated from the following files:

- kmodel.h
- kposition.cpp

6.18 DemoMtch::MtchState Class Reference

`#include <demomtch.h>`

Inheritance diagram for DemoMtch::MtchState:



Public Member Functions

- MtchState (Model ∗mod)
- ∼MtchState ()
- KMatrix actrCaps () const
- tuple< KMatrix, vector< unsigned int > > pDist (int persp) const
- void setAUtil (ReportingLevel rl)
- MtchState ∗ stepSUSN ()
- MtchState ∗ stepBCN ()

Protected Member Functions

- MtchState ∗ doSUSN (ReportingLevel rl) const
- MtchState ∗ doBCN (ReportingLevel rl) const
- bool equivNdx (unsigned int i, unsigned int j) const

Additional Inherited Members

6.18.1 Constructor & Destructor Documentation

6.18.1.1 DemoMtch::MtchState::MtchState (Model ∗ *mod*) [explicit]

6.18.1.2 DemoMtch::MtchState::∼MtchState ( )

6.18.2 Member Function Documentation

6.18.2.1 KMatrix DemoMtch::MtchState::actrCaps ( ) const

6.18.2.2 MtchState ∗ DemoMtch::MtchState::doBCN (ReportingLevel *rl*) const [protected]

6.18.2.3 MtchState ∗ DemoMtch::MtchState::doSUSN (ReportingLevel *rl*) const [protected]

6.18.2.4 bool DemoMtch::MtchState::equivNdx (unsigned int *i,* unsigned int *j*) const [protected], [virtual]

Implements KBase::State.

6.18.2.5 tuple< KMatrix, vector< unsigned int > > DemoMtch::MtchState::pDist (int *persp*) const [virtual]

Implements KBase::State.

6.18.2.6 void DemoMtch::MtchState::setAUtil (ReportingLevel *rl*)

6.18.2.7 MtchState * DemoMtch::MtchState::stepBCN ( )

6.18.2.8 MtchState * DemoMtch::MtchState::stepSUSN ( )

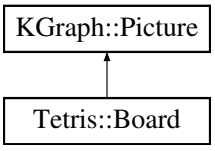
The documentation for this class was generated from the following files:

- demomtch.h
- demomtch.cpp

6.19 KGraph::Picture Class Reference

`#include <kgraph.h>`

Inheritance diagram for KGraph::Picture:



Public Member Functions

- Picture ()
- virtual ~Picture ()
- void add (Canvas *c)
- void update () const
- virtual void connect (Canvas *c)
- virtual void update (Canvas *c) const

Public Attributes

- double minX = 0
- double maxX = 1
- double minW = 1E-6
- double minY = 0
- double maxY = 1
- double minH = 1E-6

Protected Attributes

- vector< Canvas * > canvases = {}

6.19.1 Constructor & Destructor Documentation

6.19.1.1 KGraph::Picture::Picture ( )

6.19.1.2 KGraph::Picture::~Picture ( ) [virtual]

6.19.2 Member Function Documentation

6.19.2.1 void KGraph::Picture::add (Canvas * c)

6.19.2.2 void KGraph::Picture::connect (Canvas * c) [virtual]

6.19.2.3 void KGraph::Picture::update ( ) const

6.19.2.4 void KGraph::Picture::update (Canvas * c) const [virtual]

Reimplemented in Tetris::Board.

6.19.3 Member Data Documentation

6.19.3.1 vector<Canvas*> KGraph::Picture::canvases = {} [protected]

6.19.3.2 double KGraph::Picture::maxX = 1

6.19.3.3 double KGraph::Picture::maxY = 1

6.19.3.4 double KGraph::Picture::minH = 1E-6

6.19.3.5 double KGraph::Picture::minW = 1E-6

6.19.3.6 double KGraph::Picture::minX = 0

6.19.3.7 double KGraph::Picture::minY = 0

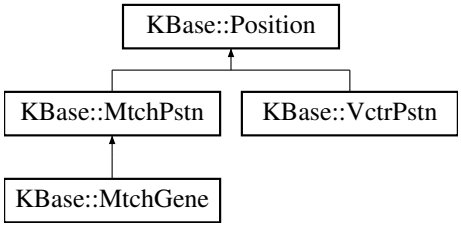
The documentation for this class was generated from the following files:

- kgraph.h
- kgraph.cpp

6.20 KBase::Position Class Reference

`#include <kmodel.h>`

Inheritance diagram for KBase::Position:



Public Member Functions

- Position ()
- virtual ~Position ()

6.20.1 Constructor & Destructor Documentation

6.20.1.1 KBase::Position::Position ( )

6.20.1.2 KBase::Position::~Position ( ) [virtual]

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

- kmodel.h
- kmodel.cpp

6.21 KBase::PRNG Class Reference

`#include <prng.h>`

Public Member Functions

- PRNG ()
- virtual ~PRNG ()
- uint64_t uniform ()
- double uniform (double a, double b)
- vector< bool > bits (unsigned int nb)
- uint64_t setSeed (uint64_t)

Protected Attributes

- mt19937_64 mt = mt19937_64()

6.21.1 Constructor & Destructor Documentation

6.21.1.1 KBase::PRNG::PRNG ( )

6.21.1.2 KBase::PRNG::~PRNG ( ) [virtual]

6.21.2 Member Function Documentation

6.21.2.1 vector< bool > KBase::PRNG::bits (unsigned int *nb*)

6.21.2.2 uint64_t KBase::PRNG::setSeed (uint64_t *s*)

6.21.2.3 uint64_t KBase::PRNG::uniform ( )

6.21.2.4 double KBase::PRNG::uniform (double *a*, double *b*)

6.21.3 Member Data Documentation

6.21.3.1 mt19937_64 KBase::PRNG::mt = mt19937_64() [protected]

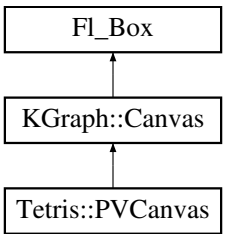
The documentation for this class was generated from the following files:

- prng.h
- prng.cpp

6.22 Tetris::PVCanvas Class Reference

`#include <pvcanvas.h>`

Inheritance diagram for Tetris::PVCanvas:



Public Member Functions

- PVCanvas (int x, int y, int w, int h, const char *l=0)
- virtual ~PVCanvas ()
- void onMove (int x, int y)
- void onDrag (int x, int y)
- void onPush (int x, int y, int b)
- void onRelease (int x, int y, int b)
- void onKeyDown (int x, int y, int k)

Protected Member Functions

- virtual void draw ()

Additional Inherited Members

6.22.1 Constructor & Destructor Documentation

6.22.1.1 Tetris::PVCanvas::PVCanvas (int *x*, int *y*, int *w*, int *h*, const char * *l* = `0`)

6.22.1.2 Tetris::PVCanvas::~PVCanvas ( ) [virtual]

6.22.2 Member Function Documentation

6.22.2.1 void Tetris::PVCanvas::draw ( ) [protected], [virtual]

6.22.2.2 void Tetris::PVCanvas::onDrag (int *x*, int *y*) [virtual]

Reimplemented from KGraph::Canvas.

6.22.2.3 void Tetris::PVCanvas::onKeyDown (int *x*, int *y*, int *k*) [virtual]

Reimplemented from KGraph::Canvas.

6.22.2.4 void Tetris::PVCanvas::onMove (int *x*, int *y*) [virtual]

Reimplemented from KGraph::Canvas.

6.22.2.5 void Tetris::PVCanvas::onPush (int *x*, int *y*, int *b*) [virtual]

Reimplemented from KGraph::Canvas.

6.22.2.6 void Tetris::PVCanvas::onRelease (int *x*, int *y*, int *b*) [virtual]

Reimplemented from KGraph::Canvas.

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

- pvcanvas.h
- pvcanvas.cpp

6.23 Tetris::Shape Class Reference

`#include <shape.h>`

Public Member Functions

- Shape ()
- Shape (TCode p)
- void setShape (TCode p)
- void setRandomShape ()
- TCode getShape () const
- char getName () const
- int x (int index) const
- int y (int index) const
- int minX () const
- int maxX () const
- int minY () const
- int maxY () const
- void showCoords () const
- Shape lrot () const
- Shape rrot () const

Public Attributes

- unsigned int idNum = 0

Static Public Attributes

- static unsigned int shapeCounter = 0

6.23.1 Constructor & Destructor Documentation

6.23.1.1 Tetris::Shape::Shape ( ) [inline]

6.23.1.2 Tetris::Shape::Shape (TCode *p*) [inline],[explicit]

6.23.2 Member Function Documentation

6.23.2.1 char Tetris::Shape::getName ( ) const [inline]

6.23.2.2 TCode Tetris::Shape::getShape ( ) const [inline]

6.23.2.3 Shape Tetris::Shape::lrot ( ) const

6.23.2.4 int Tetris::Shape::maxX ( ) const

6.23.2.5 int Tetris::Shape::maxY ( ) const

6.23.2.6 int Tetris::Shape::minX ( ) const

6.23.2.7 int Tetris::Shape::minY ( ) const

6.23.2.8 Shape Tetris::Shape::rrot ( ) const

6.23.2.9 void Tetris::Shape::setRandomShape ( )

6.23.2.10 void Tetris::Shape::setShape (TCode *p*)

6.23.2.11 void Tetris::Shape::showCoords ( ) const

6.23.2.12 int Tetris::Shape::x (int *index*) const [inline]

6.23.2.13 int Tetris::Shape::y (int *index*) const [inline]

6.23.3 Member Data Documentation

6.23.3.1 unsigned int Tetris::Shape::idNum = 0

6.23.3.2 unsigned int Tetris::Shape::shapeCounter = 0 [static]

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

- shape.h
- shape.cpp

6.24 MDemo::SQLDB Class Reference

`#include <sqlitedemo.h>`

Public Member Functions

- SQLDB (char *filename)
- virtual ~SQLDB ()
- bool open (char *filename)

- tuple< unsigned int, vector< vector< string > > > query (const char *query)
- void close ()

6.24.1 Constructor & Destructor Documentation

6.24.1.1 MDemo::SQLDB::SQLDB (char * *filename*)

6.24.1.2 MDemo::SQLDB::~SQLDB ( ) [virtual]

6.24.2 Member Function Documentation

6.24.2.1 void MDemo::SQLDB::close ( )

6.24.2.2 bool MDemo::SQLDB::open (char * *filename*)

6.24.2.3 tuple< unsigned int, vector< vector< string > > > MDemo::SQLDB::query (const char * *query*)

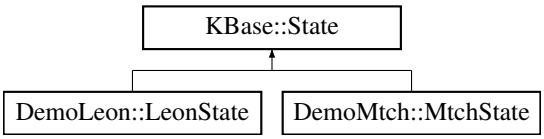
The documentation for this class was generated from the following files:

- sqlitedemo.h
- sqlitedemo.cpp

6.25 KBase::State Class Reference

`#include <kmodel.h>`

Inheritance diagram for KBase::State:



Public Member Functions

- State (Model *mod)
- virtual ~State ()
- void randomizeUtils (double minU, double maxU, double uNoise)
- void clear ()
- virtual void addPstn (Position *p)
- virtual tuple< KMatrix, vector< unsigned int > > pDist (int persp) const =0

Public Attributes

- Model * model = nullptr
- function< State *()> step = nullptr
- vector< KMatrix > aUtil = {}
- vector< Position * > pstns = {}

Protected Member Functions

- virtual bool equivNdx (unsigned int i, unsigned int j) const =0
- vector< unsigned int > testUniqueNdx (function< bool(unsigned int, unsigned int)> tfn) const
- vector< unsigned int > uniqueNdx () const

6.25.1 Constructor & Destructor Documentation

6.25.1.1 KBase::State::State (Model * *mod*) [explicit]

6.25.1.2 KBase::State::~State ( ) [virtual]

6.25.2 Member Function Documentation

6.25.2.1 void KBase::State::addPstn (Position * *p*) [virtual]

6.25.2.2 void KBase::State::clear ( )

6.25.2.3 virtual bool KBase::State::equivNdx (unsigned int *i,* unsigned int *j*) const [protected], [pure virtual]

Implemented in DemoLeon::LeonState, and DemoMtch::MtchState.

6.25.2.4 virtual tuple< KMatrix, vector<unsigned int> > KBase::State::pDist (int *persp*) const [pure virtual]

Implemented in DemoLeon::LeonState, and DemoMtch::MtchState.

6.25.2.5 void KBase::State::randomizeUtils (double *minU,* double *maxU,* double *uNoise*)

6.25.2.6 vector< unsigned int > KBase::State::testUniqueNdx (function< bool(unsigned int, unsigned int)> *tfn*) const [protected]

Given an equivalency-test, return a vector of indices to unique positions. The test might compare only the actual positions, or it might mix actual and hypothetical.

6.25.2.7 vector< unsigned int > KBase::State::uniqueNdx ( ) const [protected]

Looking only at actual current positions, return a vector of indices of unique positions

6.25.3 Member Data Documentation

6.25.3.1 vector<KMatrix> KBase::State::aUtil = {}

6.25.3.2 Model* KBase::State::model = nullptr

6.25.3.3 vector<Position*> KBase::State::pstns = {}

6.25.3.4 function<State* ()> KBase::State::step = nullptr

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

- kmodel.h
- kstate.cpp

6.26 Tetris::TApp Class Reference

`#include <tmain.h>`

Public Member Functions

- TApp (uint64_t s)
- void run ()
- virtual ~TApp ()
- void newGame ()
- void stepGame ()
- void pause ()
- void resume (double delay)
- Fl_Color color (unsigned int i) const
- void setRC (unsigned int r, unsigned int c)
- void setLevel (unsigned int lvl)
- void setRandom (bool rp)
- double setDt ()
- void applyControlState (ControlState cs)
- void applyColorScheme (ControlState cs)
- void processKey (int x, int y, int k)
- void quit ()

Public Attributes

- unsigned int level = 3
- double dt = 0.1
- PRNG * rng = nullptr
- Board * board = nullptr
- unsigned int rows = 0
- unsigned int clms = 0
- bool randomPlacement = false
- double playTime = 5*60
- double maxPlayTime = 10*60
- bool paused = true

Static Public Attributes

- static TApp * theApp = nullptr

Protected Member Functions

- unsigned int scoreFn (unsigned int clc)
- void setColorScheme ()

Protected Attributes

- unsigned int lineCount = 0
- unsigned int score = 0
- const unsigned int defaultLevel = 3
- const unsigned int defaultRows = 24
- const unsigned int defaultClms = 12
- const bool defaultRP = false
- vector< Fl_Color > colors = {}

6.26.1 Constructor & Destructor Documentation

6.26.1.1 Tetris::TApp::TApp (uint64_t *s*) [explicit]

6.26.1.2 Tetris::TApp::~TApp ( ) [virtual]

6.26.2 Member Function Documentation

6.26.2.1 void Tetris::TApp::applyColorScheme (ControlState *cs*)

6.26.2.2 void Tetris::TApp::applyControlState (ControlState *cs*)

6.26.2.3 Fl_Color Tetris::TApp::color (unsigned int *i*) const

6.26.2.4 void Tetris::TApp::newGame ( )

6.26.2.5 void Tetris::TApp::pause ( )

6.26.2.6 void Tetris::TApp::processKey (int *x,* int *y,* int *k*)

6.26.2.7 void Tetris::TApp::quit ( )

6.26.2.8 void Tetris::TApp::resume (double *delay*)

6.26.2.9 void Tetris::TApp::run ( )

6.26.2.10 unsigned int Tetris::TApp::scoreFn (unsigned int *clc*) [protected]

6.26.2.11 void Tetris::TApp::setColorScheme ( ) [protected]

6.26.2.12 double Tetris::TApp::setDt ( )

set the time between updates, given the current level

6.26.2.13 void Tetris::TApp::setLevel (unsigned int *lvl*)

6.26.2.14 void Tetris::TApp::setRandom (bool *rp*)

6.26.2.15 void Tetris::TApp::setRC (unsigned int *r,* unsigned int *c*)

6.26.2.16 void Tetris::TApp::stepGame ( )

6.26.3 Member Data Documentation

6.26.3.1 Board* Tetris::TApp::board = nullptr

6.26.3.2 unsigned int Tetris::TApp::clms = 0

6.26.3.3 vector<Fl_Color> Tetris::TApp::colors = {} [protected]

6.26.3.4 const unsigned int Tetris::TApp::defaultClms = 12 [protected]

6.26.3.5 const unsigned int Tetris::TApp::defaultLevel = 3 [protected]

6.26.3.6 const unsigned int Tetris::TApp::defaultRows = 24 [protected]

6.26.3.7 const bool Tetris::TApp::defaultRP = false [protected]

6.26.3.8 double Tetris::TApp::dt = 0.1

6.26.3.9 unsigned int Tetris::TApp::level = 3

6.26.3.10 unsigned int Tetris::TApp::lineCount = 0 [protected]

6.26.3.11 double Tetris::TApp::maxPlayTime = 10∗60

6.26.3.12 bool Tetris::TApp::paused = true

6.26.3.13 double Tetris::TApp::playTime = 5∗60

6.26.3.14 bool Tetris::TApp::randomPlacement = false

6.26.3.15 PRNG∗ Tetris::TApp::rng = nullptr

6.26.3.16 unsigned int Tetris::TApp::rows = 0

6.26.3.17 unsigned int Tetris::TApp::score = 0 [protected]

6.26.3.18 TApp ∗ TApp::theApp = nullptr [static]

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

- tmain.h
- tmain.cpp

6.27 UDemo::TargetedBV Class Reference

`#include <demo.h>`

Public Member Functions

- TargetedBV ()
- virtual ∼TargetedBV ()
- virtual void randomize (PRNG ∗rng)
- virtual TargetedBV ∗ mutate (PRNG ∗rng) const
- virtual tuple< TargetedBV ∗, TargetedBV ∗ > cross (const TargetedBV ∗g2, PRNG ∗rng) const
- virtual void show () const
- virtual bool equiv (const TargetedBV ∗g2) const
- double evaluate ()
- double tblEval (double minD, vector< double > weights, vector< BVec > tbl) const
- unsigned int hDist (BVec bv) const

Static Public Member Functions

- static void setTarget (BVec trgt)
- static BVec getTarget ()
- static void showBits (BVec bv)
- static BVec randomBV (PRNG ∗rng, unsigned int nb)

Public Attributes

- BVec bits = BVec()

Static Public Attributes

- static BVec target

6.27.1 Constructor & Destructor Documentation

6.27.1.1 UDemo::TargetedBV::TargetedBV ( )

6.27.1.2 UDemo::TargetedBV::~TargetedBV ( ) [virtual]

6.27.2 Member Function Documentation

6.27.2.1 tuple< TargetedBV *, TargetedBV * > UDemo::TargetedBV::cross (const TargetedBV * *g2*, PRNG * *rng*) const [virtual]

6.27.2.2 bool UDemo::TargetedBV::equiv (const TargetedBV * *g2*) const [virtual]

6.27.2.3 double UDemo::TargetedBV::evaluate ( )

6.27.2.4 vector< bool > UDemo::TargetedBV::getTarget ( ) [static]

6.27.2.5 unsigned int UDemo::TargetedBV::hDist (BVec *bv*) const

6.27.2.6 TargetedBV * UDemo::TargetedBV::mutate (PRNG * *rng*) const [virtual]

6.27.2.7 vector< bool > UDemo::TargetedBV::randomBV (PRNG * *rng*, unsigned int *nb*) [static]

6.27.2.8 void UDemo::TargetedBV::randomize (PRNG * *rng*) [virtual]

6.27.2.9 void UDemo::TargetedBV::setTarget (BVec *trgt*) [static]

6.27.2.10 void UDemo::TargetedBV::show ( ) const [virtual]

6.27.2.11 void UDemo::TargetedBV::showBits (BVec *bv*) [static]

6.27.2.12 double UDemo::TargetedBV::tblEval (double *minD*, vector< double > *weights*, vector< BVec > *tbl*) const

6.27.3 Member Data Documentation

6.27.3.1 BVec UDemo::TargetedBV::bits = BVec()

6.27.3.2 vector< bool > UDemo::TargetedBV::target [static]

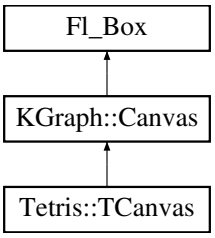
The documentation for this class was generated from the following files:

- kutils/src/demo.h
- kutils/src/demo.cpp

6.28 Tetris::TCanvas Class Reference

`#include <tcanvas.h>`

Inheritance diagram for Tetris::TCanvas:



Public Member Functions

- TCanvas (int x, int y, int w, int h, const char *l=0)
- virtual ~TCanvas ()
- void onMove (int x, int y)
- void onDrag (int x, int y)
- void onPush (int x, int y, int b)
- void onRelease (int x, int y, int b)
- void onKeyDown (int x, int y, int k)

Protected Member Functions

- virtual void draw ()

Additional Inherited Members

6.28.1 Constructor & Destructor Documentation

6.28.1.1 Tetris::TCanvas::TCanvas (int *x*, int *y*, int *w*, int *h*, const char * *l* = `0`)

6.28.1.2 Tetris::TCanvas::~TCanvas ( ) `[virtual]`

6.28.2 Member Function Documentation

6.28.2.1 void Tetris::TCanvas::draw ( ) `[protected],[virtual]`

6.28.2.2 void Tetris::TCanvas::onDrag (int *x*, int *y*) `[virtual]`

Reimplemented from KGraph::Canvas.

6.28.2.3 void Tetris::TCanvas::onKeyDown (int *x*, int *y*, int *k*) `[virtual]`

Reimplemented from KGraph::Canvas.

6.28.2.4 void Tetris::TCanvas::onMove (int *x*, int *y*) `[virtual]`

Reimplemented from KGraph::Canvas.

6.28.2.5 void Tetris::TCanvas::onPush (int *x*, int *y*, int *b*) [virtual]

Reimplemented from KGraph::Canvas.

6.28.2.6 void Tetris::TCanvas::onRelease (int *x*, int *y*, int *b*) [virtual]

Reimplemented from KGraph::Canvas.

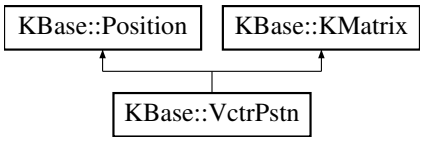
The documentation for this class was generated from the following files:

- tcanvas.h
- tcanvas.cpp

6.29 KBase::VctrPstn Class Reference

`#include <kmodel.h>`

Inheritance diagram for KBase::VctrPstn:



Public Member Functions

- VctrPstn ()
- VctrPstn (unsigned int nr, unsigned int nc)
- VctrPstn (const KMatrix &m)
- virtual ~VctrPstn ()

Additional Inherited Members

6.29.1 Constructor & Destructor Documentation

6.29.1.1 KBase::VctrPstn::VctrPstn ( )

6.29.1.2 KBase::VctrPstn::VctrPstn (unsigned int *nr*, unsigned int *nc*)

6.29.1.3 KBase::VctrPstn::VctrPstn (const KMatrix & *m*) [explicit]

6.29.1.4 KBase::VctrPstn::~VctrPstn ( ) [virtual]

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

- kmodel.h
- kposition.cpp

6.30 KBase::VHCSearch Class Reference

`#include <hcsearch.h>`

Public Member Functions

- VHCSearch ()
- virtual ~VHCSearch ()
- tuple< double, KMatrix, unsigned int, unsigned int > run (KMatrix p0, unsigned int iMax, unsigned int sMax, double sTol, double s0, double shrink, double grow, double minStep, ReportingLevel rl)

Static Public Member Functions

- static vector< KMatrix > vn1 (const KMatrix &m0, double s)
- static vector< KMatrix > vn2 (const KMatrix &m0, double s)

Public Attributes

- function< double(const KMatrix &)> eval = nullptr
- function< vector< KMatrix >const KMatrix &, double)> nghbrs = nullptr
- function< void(const KMatrix &)> report = nullptr

6.30.1 Constructor & Destructor Documentation

6.30.1.1 KBase::VHCSearch::VHCSearch ( )

6.30.1.2 KBase::VHCSearch::~VHCSearch ( ) [virtual]

6.30.2 Member Function Documentation

6.30.2.1 tuple< double, KMatrix, unsigned int, unsigned int > KBase::VHCSearch::run (KMatrix *p0,* unsigned int *iMax,* unsigned int *sMax,* double *sTol,* double *s0,* double *shrink,* double *grow,* double *minStep,* ReportingLevel *rl*)

6.30.2.2 vector< KMatrix > KBase::VHCSearch::vn1 (const KMatrix & *m0,* double *s*) [static]

6.30.2.3 vector< KMatrix > KBase::VHCSearch::vn2 (const KMatrix & *m0,* double *s*) [static]

6.30.3 Member Data Documentation

6.30.3.1 function<double(const KMatrix &)> KBase::VHCSearch::eval = nullptr

6.30.3.2 function< vector<KMatrix>const KMatrix &, double)> KBase::VHCSearch::nghbrs = nullptr

6.30.3.3 function<void (const KMatrix &)> KBase::VHCSearch::report = nullptr

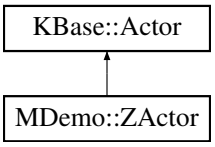
The documentation for this class was generated from the following files:

- hcsearch.h
- hcsearch.cpp

6.31 MDemo::ZActor Class Reference

`#include <zactor.h>`

Inheritance diagram for MDemo::ZActor:



Public Member Functions

- ZActor (string n, string d)
- ~ZActor ()
- double vote (unsigned int p1, unsigned int p2, const State *st) const
- virtual double vote (const Position *ap1, const Position *ap2) const
- double posUtil (const Position *ap1) const

Additional Inherited Members

6.31.1 Constructor & Destructor Documentation

6.31.1.1 MDemo::ZActor::ZActor (string *n,* string *d*)

6.31.1.2 MDemo::ZActor::~ZActor ( )

6.31.2 Member Function Documentation

6.31.2.1 double MDemo::ZActor::posUtil (const Position * *ap1*) const

6.31.2.2 double MDemo::ZActor::vote (unsigned int *p1,* unsigned int *p2,* const State * *st*) const [virtual]

Implements KBase::Actor.

6.31.2.3 double MDemo::ZActor::vote (const Position * *ap1,* const Position * *ap2*) const [virtual]

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

- zactor.h
- zactor.cpp

Chapter 7

File Documentation

7.1 board.cpp File Reference

```
#include "kutils.h"
#include "kgraph.h"
#include "prng.h"
#include "tmain.h"
#include "board.h"
#include "pvcanvas.h"
#include "tetrisUI.h"
```

Namespaces

- Tetris

7.2 board.h File Reference

```
#include "tutils.h"
#include "shape.h"
```

Classes

- class Tetris::Board

Namespaces

- Tetris

7.3 demo.cpp File Reference

```
#include "kutils.h"
#include "kmodel.h"
#include "gaopt.h"
#include "hcsearch.h"
#include "demo.h"
#include "demomtch.h"
#include "demoleon.h"
#include "sqlitedemo.h"
```

Namespaces

- MDemo

Functions

- void MDemo::demoPCE (uint64_t s, PRNG ∗rng)
- void MDemo::demoSpVSR (uint64_t s, PRNG ∗rng)
- int main (int ac, char ∗∗av)

7.3.1 Function Documentation

7.3.1.1 int main (int *ac,* char ∗∗ *av*)

7.4 demo.cpp File Reference

```
#include "demo.h"
```

Namespaces

- UDemo

Functions

- void UDemo::show (string str, const KMatrix &m, string fs)
- double UDemo::nProd (double x, double y)
- double UDemo::bsu (double d, double R)
- double UDemo::bvu (const KBase::KMatrix &d, const KBase::KMatrix &s, double R)
- void UDemo::demoThreadLambda (unsigned int n)
- void UDemo::demoThreadSynch (unsigned int n)
- void UDemo::demoMatrix (PRNG ∗rng)
- void UDemo::demoABG00 (PRNG ∗rng)
- double UDemo::eNorm (const KMatrix &a, const KMatrix &x)
- KMatrix UDemo::eUnitize (const KMatrix &a, const KMatrix &x)
- KMatrix UDemo::projEllipse (const KMatrix &a, const KMatrix &w)
- void UDemo::demoEllipseLVI (PRNG ∗rng, unsigned int n)
- tuple< KMatrix, KMatrix, KMatrix, KMatrix > UDemo::antiLemke (unsigned int n)
- void UDemo::demoAntiLemke (PRNG ∗rng, unsigned int n)
- void UDemo::demoEllipse (PRNG ∗rng)

- void UDemo::demoGA (PRNG *rng)
- void UDemo::demoGHC (PRNG *rng)
- void UDemo::demoVHC00 (PRNG *rng)
- void UDemo::demoVHC01 (PRNG *rng)
- void UDemo::demoVHC02 (PRNG *rng)
- void UDemo::demoVHC03 (PRNG *rng)
- void UDemo::parallelMatrixMult (PRNG *rng)
- int main (int ac, char **av)

7.4.1 Function Documentation

7.4.1.1 int main (int *ac*, char ** *av*)

7.5 demo.h File Reference

```
#include <assert.h>
#include <chrono>
#include <cstring>
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include "kutils.h"
#include "prng.h"
#include "kmatrix.h"
#include "kmodel.h"
```

Namespaces

- MDemo

7.6 demo.h File Reference

```
#include "kutils.h"
#include "prng.h"
#include "kmatrix.h"
#include "gaopt.h"
#include "hcsearch.h"
#include "vimcp.h"
```

Classes

- class UDemo::TargetedBV

Namespaces

- UDemo

Typedefs

- typedef vector< bool > UDemo::BVec

Functions

- double UDemo::eNorm (const KMatrix &a, const KMatrix &x)
- KMatrix UDemo::eUnitize (const KMatrix &a, const KMatrix &x)
- KMatrix UDemo::projEllipse (const KMatrix &a, const KMatrix &w)
- void UDemo::demoEllipseLVI (PRNG *rng, unsigned int n)
- void UDemo::demoAntiLemke (PRNG *rng, unsigned int n)

7.7 demoleon.cpp File Reference

```
#include "demoleon.h"
```

Namespaces

- DemoLeon

Functions

- LeonModel * DemoLeon::demoSetup (unsigned int numFctr, unsigned int numCGrp, unsigned int numSect, uint64_t s, PRNG *rng)
- void DemoLeon::demoEUEcon (uint64_t s, unsigned int numF, unsigned int numG, unsigned int numS, PRNG *rng)
- void DemoLeon::demoMaxEcon (uint64_t s, unsigned int numF, unsigned int numG, unsigned int numS, PRNG *rng)
- int main (int ac, char **av)

Variables

- const double DemoLeon::TolIFD = 1E-6

7.7.1 Function Documentation

7.7.1.1 int main (int *ac,* char ** *av*)

7.8 demoleon.h File Reference

```
#include <assert.h>
#include <chrono>
#include <cstring>
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <tuple>
#include <vector>
#include "kutils.h"
#include "prng.h"
#include "kmatrix.h"
#include "gaopt.h"
#include "hcsearch.h"
#include "kmodel.h"
```

Classes

- class DemoLeon::LeonActor
- class DemoLeon::LeonState
- class DemoLeon::LeonModel

Namespaces

- DemoLeon

Functions

- LeonModel * DemoLeon::demoSetup (unsigned int numFctr, unsigned int numCGrp, unsigned int numSect, uint64_t s, PRNG *rng)
- void DemoLeon::demoEUEcon (uint64_t s, PRNG *rng)
- void DemoLeon::demoMaxEcon (uint64_t s, PRNG *rng)

7.9 demomtch.cpp File Reference

```
#include "demomtch.h"
```

Namespaces

- DemoMtch

Functions

- bool DemoMtch::equivMtchPstn (const MtchPstn &mp1, const MtchPstn &mp2)
- void DemoMtch::showMtchPstn (const MtchPstn &mp)
- bool DemoMtch::stableMtchState (unsigned int iter, const State *s1)
- void DemoMtch::demoDivideSweets (uint64_t s, PRNG *rng)
- void DemoMtch::demoMaxSupport (uint64_t s, PRNG *rng)

- void DemoMtch::demoMtchSUSN (uint64_t s, PRNG *rng)
- void DemoMtch::multiMtchSUSN (uint64_t s, PRNG *rng)
- bool DemoMtch::oneMtchSUSN (uint64_t s, PRNG *rng)
- int main (int ac, char **av)

7.9.1 Function Documentation

7.9.1.1 int main (int *ac,* char ** *av*)

7.10 demomtch.h File Reference

```
#include <assert.h>
#include <chrono>
#include <cstring>
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <tuple>
#include <vector>
#include "kutils.h"
#include "prng.h"
#include "kmatrix.h"
#include "gaopt.h"
#include "hcsearch.h"
#include "kmodel.h"
```

Classes

- class DemoMtch::MtchActor
- class DemoMtch::MtchState
- class DemoMtch::MtchModel

Namespaces

- DemoMtch

Functions

- void DemoMtch::demoDivideSweets (uint64_t s, PRNG *rng)
- void DemoMtch::demoMaxSupport (uint64_t s, PRNG *rng)
- void DemoMtch::demoMtchSUSN (uint64_t s, PRNG *rng)
- void DemoMtch::multiMtchSUSN (uint64_t s, PRNG *rng)
- bool DemoMtch::oneMtchSUSN (uint64_t s, PRNG *rng)
- void DemoMtch::showMtchPstn (const MtchPstn &mp)
- bool DemoMtch::stableMtchState (unsigned int iter, const State *s1)

7.11 gaopt.cpp File Reference

```
#include <assert.h>
#include <iostream>
#include <tuple>
#include "kutils.h"
#include "prng.h"
#include "gaopt.h"
```

Namespaces

- KBase

Functions

- unsigned int KBase::crossSite (PRNG *rng, unsigned int nc)

7.12 gaopt.h File Reference

```
#include <assert.h>
#include <chrono>
#include <functional>
#include <iostream>
#include <string>
#include <tuple>
#include <vector>
#include "prng.h"
#include "kutils.h"
```

Classes

- class KBase::GAOpt< GAP >

Namespaces

- KBase

Functions

- unsigned int KBase::crossSite (PRNG *rng, unsigned int nc)

7.13 hcsearch.cpp File Reference

```
#include "kutils.h"
#include "hcsearch.h"
```

Namespaces

- KBase

7.14 hcsearch.h File Reference

```
#include <functional>
#include <iostream>
#include <tuple>
#include <vector>
#include "kutils.h"
#include "kmatrix.h"
```

Classes

- class KBase::VHCSearch
- class KBase::GHCSearch< HCP >

Namespaces

- KBase

7.15 kgraph.cpp File Reference

```
#include <FL/Fl.H>
#include <FL/Enumerations.H>
#include "kgraph.h"
```

Namespaces

- KGraph

7.16 kgraph.h File Reference

```
#include <assert.h>
#include <chrono>
#include <cstdint>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <functional>
#include <future>
#include <math.h>
#include <memory>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <thread>
#include <tuple>
#include <vector>
#include <FL/Fl.H>
#include <FL/Fl_Box.H>
#include <FL/Fl_Double_Window.H>
#include <FL/fl_draw.H>
#include <FL/Fl_Group.H>
```

Classes

- class KGraph::CoordMap
- class KGraph::Canvas
- class KGraph::Picture

Namespaces

- KGraph

7.17 kmatrix.cpp File Reference

```
#include <assert.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <iostream>
#include <string.h>
#include <vector>
#include "prng.h"
#include "kmatrix.h"
```

Namespaces

- KBase

Functions

- KMatrix KBase::trans (const KMatrix &m1)
- double KBase::norm (const KMatrix &m1)
- double KBase::sum (const KMatrix &m1)
- double KBase::mean (const KMatrix &m1)
- double KBase::stdv (const KMatrix &m1)
- double KBase::maxAbs (const KMatrix &m)
- tuple< unsigned int, unsigned int > KBase::ndxMaxAbs (const KMatrix &m)
- double KBase::lCorr (const KMatrix &m1, const KMatrix &m2)
- double KBase::dot (const KMatrix &m1, const KMatrix &m2)
- KMatrix KBase::operator+ (const KMatrix &m1, double x)
- KMatrix KBase::operator- (const KMatrix &m1, double x)
- bool KBase::sameShape (const KMatrix &m1, const KMatrix &m2)
- KMatrix KBase::operator+ (const KMatrix &m1, const KMatrix &m2)
- KMatrix KBase::operator- (const KMatrix &m1, const KMatrix &m2)
- KMatrix KBase::operator* (double x, const KMatrix &m1)
- KMatrix KBase::operator/ (const KMatrix &m1, double x)
- KMatrix KBase::operator* (const KMatrix &m1, const KMatrix &m2)
- KMatrix KBase::inv (const KMatrix &m)
- KMatrix KBase::iMat (unsigned int n)
- KMatrix KBase::makePerp (const KMatrix &x, const KMatrix &p)
- KMatrix KBase::joinH (const KMatrix &mL, const KMatrix &mR)
- KMatrix KBase::joinV (const KMatrix &mT, const KMatrix &mB)

7.18 kmatrix.h File Reference

```
#include <cstdint>
#include <functional>
#include <tuple>
#include <vector>
#include "kutils.h"
```

Classes

- class KBase::KMatrix

Namespaces

- KBase

Functions

- KMatrix KBase::trans (const KMatrix &m1)
- double KBase::norm (const KMatrix &m1)
- double KBase::sum (const KMatrix &m1)
- double KBase::mean (const KMatrix &m1)
- double KBase::stdv (const KMatrix &m1)
- double KBase::maxAbs (const KMatrix &m)
- tuple< unsigned int, unsigned int > KBase::ndxMaxAbs (const KMatrix &m)
- double KBase::dot (const KMatrix &m1, const KMatrix &m2)

- double KBase::lCorr (const KMatrix &m1, const KMatrix &m2)
- KMatrix KBase::inv (const KMatrix &m)
- KMatrix KBase::iMat (unsigned int n)
- KMatrix KBase::makePerp (const KMatrix &x, const KMatrix &p)
- KMatrix KBase::joinH (const KMatrix &mL, const KMatrix &mR)
- KMatrix KBase::joinV (const KMatrix &mT, const KMatrix &mB)
- KMatrix KBase::operator+ (const KMatrix &m1, const KMatrix &m2)
- KMatrix KBase::operator+ (const KMatrix &m1, double x)
- KMatrix KBase::operator- (const KMatrix &m1, const KMatrix &m2)
- KMatrix KBase::operator- (const KMatrix &m1, double x)
- KMatrix KBase::operator∗ (double x, const KMatrix &m1)
- KMatrix KBase::operator/ (const KMatrix &m1, double x)
- bool KBase::sameShape (const KMatrix &m1, const KMatrix &m2)
- KMatrix KBase::operator∗ (const KMatrix &m1, const KMatrix &m2)

7.19 kmodel.cpp File Reference

```
#include <assert.h>
#include <iostream>
#include "kmodel.h"
```

Namespaces

- KBase

Functions

- string KBase::vrName (VotingRule vr)
- string KBase::tpcName (ThirdPartyCommit tpc)

7.20 kmodel.h File Reference

```
#include <sqlite3.h>
#include "kutils.h"
#include "kmatrix.h"
#include "prng.h"
```

Classes

- class KBase::Position
- class KBase::VctrPstn
- class KBase::MtchPstn
- class KBase::MtchGene
- class KBase::Model
- class KBase::State
- class KBase::Actor

Namespaces

- KBase

Enumerations

- enum KBase::VotingRule : char {
  KBase::VotingRule::Binary, KBase::VotingRule::PropBin, KBase::VotingRule::Proportional, KBase::Voting↩
  Rule::PropCbc,
  KBase::VotingRule::Cubic }
- enum KBase::ThirdPartyCommit { KBase::ThirdPartyCommit::NoCommit, KBase::ThirdPartyCommit::Semi↩
  Commit, KBase::ThirdPartyCommit::FullCommit }

Functions

- string KBase::vrName (VotingRule vr)
- string KBase::tpcName (ThirdPartyCommit tpc)
- vector< MtchPstn > KBase::uniqueMP (vector< MtchPstn > mps)
- bool operator== (const KBase::MtchPstn &mp1, const KBase::MtchPstn &mp2)

7.20.1 Function Documentation

7.20.1.1 bool operator== (const KBase::MtchPstn & *mp1,* const KBase::MtchPstn & *mp2*)

7.21 kposition.cpp File Reference

```
#include <iostream>
#include "gaopt.h"
#include "kmodel.h"
```

Namespaces

- KBase

Functions

- bool operator== (const KBase::MtchPstn &mp1, const KBase::MtchPstn &mp2)
- vector< MtchPstn > KBase::uniqueMP (vector< MtchPstn > mps)

7.21.1 Function Documentation

7.21.1.1 bool operator== (const KBase::MtchPstn & *mp1,* const KBase::MtchPstn & *mp2*)

7.22 kstate.cpp File Reference

```
#include "kmodel.h"
```

Namespaces

- KBase

7.23 kutils.cpp File Reference

```
#include <assert.h>
#include <iostream>
#include <tuple>
#include "kutils.h"
#include "prng.h"
```

Namespaces

- KBase

Functions

- double KBase::sqr (const double x)
- double KBase::qrtc (const double x)
- std::chrono::time_point< std::chrono::system_clock > KBase::displayProgramStart ()
- void KBase::displayProgramEnd (std::chrono::time_point< std::chrono::system_clock > st)
- char * KBase::newChars (unsigned int len)
- double KBase::rescale (double x, double x0, double x1, double y0, double y1)

7.24 kutils.h File Reference

```
#include <assert.h>
#include <chrono>
#include <cstdint>
#include <cstdlib>
#include <cstring>
#include <ctime>
#include <iomanip>
#include <iostream>
#include <functional>
#include <future>
#include <math.h>
#include <memory>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <thread>
#include <tuple>
#include <vector>
```

Classes

- class KBase::KException

Namespaces

- KBase

Enumerations

- enum KBase::ReportingLevel {
  KBase::ReportingLevel::Silent, KBase::ReportingLevel::Low, KBase::ReportingLevel::Medium, KBase::↩
  ReportingLevel::High,
  KBase::ReportingLevel::Debugging }

Functions

- std::chrono::time_point< std::chrono::system_clock > KBase::displayProgramStart ()
- void KBase::displayProgramEnd (std::chrono::time_point< std::chrono::system_clock > st)
- char * KBase::newChars (unsigned int len)
- double KBase::rescale (double x, double x0, double x1, double y0, double y1)
- template<typename T >
  T KBase::popBack (vector< T > &v)
- double KBase::sqr (const double x)
- double KBase::qrtc (const double x)

7.25 prng.cpp File Reference

```
#include <assert.h>
#include "prng.h"
```

Namespaces

- KBase

Functions

- uint64_t KBase::qTrans (uint64_t s)

7.26 prng.h File Reference

```
#include <cstdint>
#include <random>
#include "kutils.h"
```

Classes

- class KBase::PRNG

Namespaces

- KBase

Functions

- uint64_t KBase::qTrans (uint64_t s)

7.27 pvcanvas.cpp File Reference

```
#include <string.h>
#include "tutils.h"
#include "pvcanvas.h"
#include "tmain.h"
```

Namespaces

- Tetris

7.28 pvcanvas.h File Reference

```
#include "kutils.h"
#include "kgraph.h"
```

Classes

- class Tetris::PVCanvas

Namespaces

- Tetris

7.29 shape.cpp File Reference

```
#include "kutils.h"
#include "shape.h"
#include "tmain.h"
```

Namespaces

- Tetris

7.30 shape.h File Reference

Classes

- class Tetris::Shape

Namespaces

- Tetris

Enumerations

- enum Tetris::TCode {
  Tetris::N = 0, Tetris::I, Tetris::J, Tetris::L,
  Tetris::O, Tetris::S, Tetris::T, Tetris::Z }

7.31 sqlitedemo.cpp File Reference

```
#include <cstdlib>
#include <string>
#include <iostream>
#include <stdio.h>
#include <sqlite3.h>
#include "sqlitedemo.h"
```

Namespaces

- MDemo

Functions

- void MDemo::demoDBObject ()

7.32 sqlitedemo.h File Reference

```
#include <cstdlib>
#include <iostream>
#include <stdio.h>
#include <string>
#include <sqlite3.h>
#include <tuple>
#include <vector>
#include "kutils.h"
#include "prng.h"
#include "kmatrix.h"
```

Classes

- class MDemo::SQLDB

Namespaces

- MDemo

Functions

- void MDemo::demoDBObject ()

7.33 tcanvas.cpp File Reference

```
#include <string.h>
#include "tutils.h"
#include "tcanvas.h"
#include "tmain.h"
#include "tetrisUI.h"
```

Namespaces

- Tetris

7.34 tcanvas.h File Reference

```
#include "kutils.h"
#include "kgraph.h"
```

Classes

- class Tetris::TCanvas

Namespaces

- Tetris

7.35 tmain.cpp File Reference

```
#include "kutils.h"
#include "tutils.h"
#include "prng.h"
#include "tmain.h"
#include "tetrisUI.h"
```

Namespaces

- Tetris

Functions

- char * Tetris::newChar (unsigned int buffLen)
- void Tetris::tetrisTimer (void *)
- int main (int ac, char **av)

7.35.1 Function Documentation

7.35.1.1 int main (int *ac*, char ** *av*)

7.36 tmain.h File Reference

```
#include "tutils.h"
#include "tcanvas.h"
#include "board.h"
```

Classes

- class Tetris::ControlState
- class Tetris::TApp

Namespaces

- Tetris

Enumerations

- enum Tetris::SchemeShapes { Tetris::SS_GameBoy, Tetris::SS_Gerasimov, Tetris::SS_Sega, Tetris::SS_SovietMG, Tetris::SS_TetrisCo }
- enum Tetris::SchemeWindows { Tetris::SW_Black, Tetris::SW_White, Tetris::SW_Beige }

Functions

- char * Tetris::newChar (unsigned int buffLen)

7.37 tutils.h File Reference

```
#include <assert.h>
#include <chrono>
#include <cstdint>
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <functional>
#include <future>
#include <math.h>
#include <memory>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <thread>
#include <tuple>
#include <vector>
#include <FL/Enumerations.H>
#include "kutils.h"
#include "prng.h"
#include "kgraph.h"
```

Namespaces

- Tetris

Functions

- void Tetris::tetrisTimer (void *)

7.38 vimcp.cpp File Reference

```
#include "vimcp.h"
```

Namespaces

- KBase

Functions

- KMatrix KBase::projPos (const KMatrix &w)
- KMatrix KBase::projBox (const KMatrix &lb, const KMatrix &ub, const KMatrix &w)
- tuple< KMatrix, unsigned int, KMatrix > KBase::viABG (const KMatrix &xInit, function< KMatrix(const K↩ Matrix &x)> F, function< KMatrix(const KMatrix &x)> P, double beta, double thresh, unsigned int iMax, bool extra)
- tuple< KMatrix, unsigned int, KMatrix > KBase::viBSHe96 (const KMatrix &M, const KMatrix &q, function< KMatrix(const KMatrix &)> pK, KMatrix u0, const double eps, const unsigned int iMax)

7.39 vimcp.h File Reference

```
#include <assert.h>
#include <functional>
#include <iostream>
#include <tuple>
#include "kutils.h"
#include "kmatrix.h"
#include "prng.h"
```

Namespaces

- KBase

Functions

- tuple< KMatrix, KMatrix, KMatrix, KMatrix > KBase::antiLemke (unsigned int n)
- KMatrix KBase::projPos (const KMatrix &w)
- KMatrix KBase::projBox (const KMatrix &lb, const KMatrix &ub, const KMatrix &w)

- tuple< KMatrix, unsigned int, KMatrix > KBase::viABG (const KMatrix &xInit, function< KMatrix(const K↩ Matrix &x)> F, function< KMatrix(const KMatrix &x)> P, double beta, double thresh, unsigned int iMax, bool extra)
- tuple< KMatrix, unsigned int, KMatrix > KBase::viBSHe96 (const KMatrix &M, const KMatrix &q, function< KMatrix(const KMatrix &)> pK, KMatrix u0, const double eps, const unsigned int iMax)

7.40 zactor.cpp File Reference

```
#include "kutils.h"
#include "kmodel.h"
#include "demo.h"
#include "zactor.h"
```

Namespaces

- MDemo

7.41 zactor.h File Reference

```
#include <assert.h>
#include <chrono>
#include <cstring>
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <tuple>
#include <vector>
#include "kutils.h"
#include "prng.h"
#include "kmatrix.h"
#include "gaopt.h"
#include "kmodel.h"
```

Classes

- class MDemo::ZActor

Namespaces

- MDemo