```
std::map< char, Node</li>
                        < char > * > nodes
                        Node < char > * starterNode
                        std::vector< Edge</li>
                         char > * > edges
                        + Model()
                        + char * RandomWalk(Markov
                        ::Random::RandomEngine
                         *randomEngine, int minSetting,
                         int maxSetting, char *buffer)
                        + void AdjustEdge(const
                         char *payload, long
                         int occurrence)
                        + bool Import(std::ifstream *)
                        + bool Import(const char
                         *filename)
                        + bool Export(std::ofstream *)
                        + bool Export(const char
                         *filename)
                        + Node< char > * StarterNode()
                        + std::vector< Edge<
                         char > * > * Edges()
                        + std::map< char, Node
                        < char > * > * Nodes()
                        + void OptimizeEdgeOrder()
                         Markov::API::MarkovPasswords
                 - std::ifstream * datasetFile
                 std::ofstream * modelSavefile
                 std::ofstream * outputFile
                 + MarkovPasswords()
                 + MarkovPasswords(const
                  char *filename)
                 + std::ifstream * OpenDataset
                 File(const char *filename)
                 + void Train(const char
                  *datasetFileName, char
                  delimiter, int threads)
                 + std::ofstream * Save
                 (const char *filename)
                 + void Generate(unsigned
                  long int n, const char
                  *wordlistFileName, int
                  minLen=6, int maxLen=12,
                  int threads=20)
                 + void Buff(const char
                  *str, double multiplier,
                  bool bDontAdjustSelfLoops
                 =true, bool bDontAdjustExtendedLoops=false)

    void TrainThread(Markov

                 ::API::Concurrency::ThreadShared
                 ListHandler *listhandler. char
                  delimiter)

    void GenerateThread

                 (std::mutex *outputLock,
                  unsigned long int n, std
                 ::ofstream *wordlist, int
                  minLen, int maxLen)
                            Markov::API::ModelMatrix
                     # char ** edgeMatrix
                     # long int ** valueMatrix
                     # int matrixSize
                     # char * matrixIndex
                     # long int * totalEdgeWeights
                     # bool ready
                     + ModelMatrix()
                     + bool ConstructMatrix()
                     + void DumpJSON()
                     + int FastRandomWalk
                     (unsigned long int n,
                     const char *wordlistFileName,
                     int minLen=6, int maxLen
                     =12, int threads=20, bool bFileIO=true)
                     + void Import(const char
                     *filename)
                     + void Train(const char
                     *datasetFileName, char
                     delimiter, int threads)
                     # int FastRandomWalk
                     (unsigned long int n,
                     std::ofstream *wordlist,
                     int minLen=6, int maxLen
                     =12, int threads=20, bool
                     bFileIO=true)
                     # void FastRandomWalkPartition
                     (std::mutex *mlock, std::
                     ofstream *wordlist, unsigned
                     long int n, int minLen, int
                     maxLen, bool bFileIO, int threads)
                     # void FastRandomWalkThread
                     (std::mutex *mlock, std
                     ::ofstream *wordlist, unsigned
                     long int n, int minLen, int
                     maxLen, int id, bool bFileIO)
                     # bool DeallocateMatrix()
                                          Python.Markopy.ModelMatrix
                                          + def FastRandomWalk
                                          (int count, str wordlist,
                                           int minlen, int maxlen)
Markov::API::CUDA::
          CUDAModelMatrix

    char * device edgeMatrix

long int * device valueMatrix
char * device matrixIndex

    long int * device totalEdge

Weights
- char ** device_outputBuffer
- char ** outputBuffer
char * flatEdgeMatrix
long int * flatValueMatrix
- int cudaBlocks

    int cudaThreads

    int iterationsPerKernelThread

    long int totalOutputPerSync

    long int totalOutputPerKernel

    int numberOfPartitions

                                              Python.Markopy.ModelMatrixCLI

    int cudaGridSize

- int cudaMemPerGrid
                                              + model
                                              + fileIO

    long int cudaPerKernelAllocationSize

- int alternating Kernels
                                              + def init (self,
- unsigned long ** device
                                              bool add_help=True)
+ def add_arguments(self)
seeds
- cudaStream t * cudastreams
                                              + def init post arguments
    host void MigrateMatrix()
                                              (self)
    host void FlattenMatrix()
                                              - def _generate(self,
    host__ void FastRandom
                                              str wordlist)
Walk(unsigned long int
n, const char *wordlistFileName,
int minLen, int maxLen, bool
bFileIO, bool bInfinite)
    host char * AllocVRAMOutput
Buffer(long int n, long int
singleGenMaxLen, long int CUDAKernel
GridSize, long int sizePerGrid)
# host void LaunchAsync
Kernel(int kernelID, int
minLen, int maxLen)
# host void prepKernel
MemoryChannel(int numberOfStreams)
# __host__ void GatherAsync
KernelOutput(int kernelID,
bool bFileIO, std::ofstream
&wordlist)
                                               Python.Markopy.MarkopyCLI
        Python.CudaMarkopy.CudaModel
                                                + args
                    MatrixCLI
                                                + cli
         + model
                                                + def init (self,
         + bInfinite
                                                add help=False)
                                                + def add arguments(self)
         + def init (self)
                                                + def help(self)
         + def add_arguments(self)
                                               + def parse(self)
         + def init post arguments
                                               + def init post arguments
         (self)
         def _generate(self,
                                                + def parse fail(self)
         str wordlist)
                                                + def process(self)
                                                + def stub(self)
                        Python.CudaMarkopy.CudaMarkopyCLI
                        + args
                        + cli
                        + None init (self)
                        + def help(self)
```

+ def parse(self)
+ def parse fail(self)

Markov::Model < char >