**Includes**

Converter

#include<opencv2/core/core.hpp>

#include<Eigen/Dense>

#include"Thirdparty/g2o/g2o/types/types\_six\_dof\_expmap.h"

#include"Thirdparty/g2o/g2o/types/types\_seven\_dof\_expmap.h"

Frame

#include<vector>

#include "MapPoint.h"

#include "Thirdparty/DBoW2/DBoW2/BowVector.h"

#include "Thirdparty/DBoW2/DBoW2/FeatureVector.h"

#include "ORBVocabulary.h"

#include "KeyFrame.h"

#include "ORBextractor.h"

#include <opencv2/opencv.hpp>

FrameDrawer

#include "Tracking.h"

#include "MapPoint.h"

#include "Map.h"

#include<opencv2/core/core.hpp>

#include<opencv2/features2d/features2d.hpp>

#include<mutex>

Initializer

#include<opencv2/opencv.hpp>

#include "Frame.h"

KeyFrame

#include "MapPoint.h"

#include "Thirdparty/DBoW2/DBoW2/BowVector.h"

#include "Thirdparty/DBoW2/DBoW2/FeatureVector.h"

#include "ORBVocabulary.h"

#include "ORBextractor.h"

#include "Frame.h"

#include "KeyFrameDatabase.h"

#include <mutex>

KeyFrameDatabase

#include <vector>

#include <list>

#include <set>

#include "KeyFrame.h"

#include "Frame.h"

#include "ORBVocabulary.h"

#include<mutex>

LocalMapping

#include "KeyFrame.h"

#include "Map.h"

#include "LoopClosing.h"

#include "Tracking.h"

#include "KeyFrameDatabase.h"

#include <mutex>

LoopClosing

#include "KeyFrame.h"

#include "LocalMapping.h"

#include "Map.h"

#include "ORBVocabulary.h"

#include "Tracking.h"

#include "KeyFrameDatabase.h"

#include <thread>

#include <mutex>

#include "Thirdparty/g2o/g2o/types/types\_seven\_dof\_expmap.h"

Map

#include "MapPoint.h"

#include "KeyFrame.h"

#include <set>

#include <mutex>

MapDrawer

#include"Map.h"

#include"MapPoint.h"

#include"KeyFrame.h"

#include<pangolin/pangolin.h>

#include<mutex>

MapPoint

#include"KeyFrame.h"

#include"Frame.h"

#include"Map.h"

#include<opencv2/core/core.hpp>

#include<mutex>

Optimizer

#include "Map.h"

#include "MapPoint.h"

#include "KeyFrame.h"

#include "LoopClosing.h"

#include "Frame.h"

#include "Thirdparty/g2o/g2o/types/types\_seven\_dof\_expmap.h"

ORBextractor

#include <vector>

#include <list>

#include <opencv/cv.h>

ORBmatcher

#include<vector>

#include<opencv2/core/core.hpp>

#include<opencv2/features2d/features2d.hpp>

#include"MapPoint.h"

#include"KeyFrame.h"

#include"Frame.h"

ORBVocabulary

#include"Thirdparty/DBoW2/DBoW2/FORB.h"

#include"Thirdparty/DBoW2/DBoW2/TemplatedVocabulary.h"

PnPsolver

#include <opencv2/core/core.hpp>

#include "MapPoint.h"

#include "Frame.h"

Sim3Solver

#include <opencv2/opencv.hpp>

#include <vector>

#include "KeyFrame.h"

System

#include<string>

#include<thread>

#include<opencv2/core/core.hpp>

#include "Tracking.h"

#include "FrameDrawer.h"

#include "MapDrawer.h"

#include "Map.h"

#include "LocalMapping.h"

#include "LoopClosing.h"

#include "KeyFrameDatabase.h"

#include "ORBVocabulary.h"

#include "Viewer.h"

Tracking

#include<opencv2/core/core.hpp>

#include<opencv2/features2d/features2d.hpp>

#include"Viewer.h"

#include"FrameDrawer.h"

#include"Map.h"

#include"LocalMapping.h"

#include"LoopClosing.h"

#include"Frame.h"

#include "ORBVocabulary.h"

#include"KeyFrameDatabase.h"

#include"ORBextractor.h"

#include "Initializer.h"

#include "MapDrawer.h"

#include "System.h"

#include <mutex>

Viewer

#include "FrameDrawer.h"

#include "MapDrawer.h"

#include "Tracking.h"

#include "System.h"

#include <mutex>

Main

#include<iostream>

#include<algorithm>

#include<fstream>

#include<chrono>

#include<opencv2/core/core.hpp>

#include<System.h>

**Actual Dependencies**

Converter

Frame

FrameDrawer

Initializer

KeyFrame

KeyFrameDatabase

LocalMapping

LoopClosing

Map

MapPoint

Keyframe

MapDrawer

MapPoint

Optimizer

ORBextractor

None

ORBmatcher

None

ORBVocabulary

None

PnPSolver

None

Sim3Solver

KeyFrame

System

Tracking

Viewer

**What are the downsides to forward declarations?**

The [Google style guide](https://google.github.io/styleguide/cppguide.html#Forward_Declarations) recommends against using forward declarations, and for good reasons:

* If someone forward declares something from namespace std, then your code exhibits undefined behavior (but will likely work).
* Forward declarations can easily become redundant when an API is changed such that it’s unavoidable to know the full size and alignment of a dependent type. You may end up with both a #include and a forward declaration in your header in this case.
* There are some rare cases where your code may behave differently. You may not be bringing in additional function overloads or template specializations that you previously relied upon, or you lose inheritance information which can cause a different overload or specialization to be called in the first place.