1 Package cpp2latex

1.1 Summary

This package `cpp2latex` converts a `C++` file into LaTeX code, to be included in a LaTeX document.
It creates a box with a title and does some syntax highlighting; a canonical "Hello World" `C++` file looks like this:

main.cc

```cpp
#include <iostream>

int main(int argc, char *argv[])
{
  std::cout << "Hello World." << std::endl;
  return 0;
}
```

Table 1.1: main.cc

1.2 Usage

You include the generated file into your LaTeX document like this:

```
\begin{table}[ht]
  \rowcolors{0}{gray!15}{gray!5}
  \begin{center}
    \scalebox{1.1}{
      \tcbox[title=main.cc] {
        \input{test/main.cc.tex}
      }
    }
  \end{center}
  \caption{main.cc}
\end{table}
```

And you generate the file like this:

```
cat test/main.cc | bin/cpp2latex > test/main.cc.tex
```

1.3 Build

This package depends on reflex, a modern C++ replacement of gnu flex. See https://www.genivia.com/doc/reflex/html/

Please check/adjust the `Makefile` to reflect how "reflex" is installed on your system.

To build the package, you use make:

```
make
```

You can also generate an example pdf file with:

```
make example
```

This requires that "pdflatex" is installed.

it "installs" the binary in `/bin/cpp2latex`; if you wish you may run

```
make install
```

creates a symlink in `\usr\local\bin`.

2 Examples

Euler14.h

```
#pragma once

#include "Ints.h"

class Euler14
{
 public:
  Euler14(int argc, char *argv[]);

  void exec();

 private:
  const int m_argc;
  char ** m_argv;

  U64 m_start;
  U64 m_stop;

  bool m_show_sequence = false;
  bool m_show_detailresult = false;
};
```

Table 2.1: Euler14.h

bench.cc

```
#include "benchmark/benchmark.h"
#include <cmath>
// #include "Isqrt.h"
#include <iostream>

/*
static void BM_Test(benchmark::State & state)
{
  unsigned long sum = 0;
  for (auto _ : state)
    {
      for( unsigned int i = 0; i < 1000000; ++i )
        sum += Math::sqrt(i);
    }
  std::cout << sum << "\n";
}
*/

static void BM2_Test(benchmark::State & state)
{
  unsigned long sum = 0;
  for (auto _ : state)
    {
      for( unsigned int i = 0; i < 1000000; ++i )
        sum += (unsigned)std::sqrt(i);
    }
  std::cout << sum << "\n";
}

//BENCHMARK(BM_Test);
BENCHMARK(BM2_Test);
```

Table 2.2: bench.cc

dbinit.cc

```cpp
#include "DB_Initialize.h"

#include "DBU_FirstNamesMale.h"
#include "DBU_FirstNamesFemale.h"
#include "DBU_FamilyNames.h"
#include "DB_Students.h"
#include "DB_Teachers.h"
#include "DBA_StudentTeachers.h"

namespace
{
   struct Context
   {
         static const size_t C1 = 1777;
         static const size_t C2 = 2777;
         static const size_t C3 = 3217;
         size_t maleIdx = 7;
         size_t femaleIdx = 11;
         size_t familyIdx = 13;
   };

   void getNextName( Context & ctx,          ///< [inout]
                     size_t & n1,
                     size_t & n2,
                     bool & is_male
                   )
   {
      static const size_t males = S_DBU_FirstNamesMale::getInstance().getSize();
      static const size_t females = S_DBU_FirstNamesFemale::getInstance().getSize();
      static const size_t numFamilyNames = S_DBU_FamilyNames::getInstance().getSize();

      is_male = ctx.familyIdx % 2 == 0;
      ctx.maleIdx = (ctx.maleIdx + Context::C1 + rand() % 100) % males;
      ctx.femaleIdx = (ctx.femaleIdx + Context::C2 + rand() % 99) % females;
      ctx.familyIdx = (ctx.familyIdx + Context::C3 + rand() % 98) % numFamilyNames;

      n1 = is_male ? ctx.maleIdx : ctx.femaleIdx;
      n2 = ctx.familyIdx;
   }
}

void DB::initStudents( const size_t numOfStudents )
{
   size_t s = 0;
   Context ctx;

   while( s < numOfStudents )
   {
      size_t i1,i2;
      bool is_male;
      getNextName(ctx, i1, i2, is_male);

      S_DB_Students::getInstance().add(
        { is_male,
          i1,
          i2,
          42
        });

      ++s;
   }
}

void DB::initTeachers( const size_t numOfTeachers )
{
   size_t s = 0;
   Context ctx;

   while( s < numOfTeachers )
   {
      size_t i1,i2;
      bool is_male;
      getNextName(ctx, i1, i2, is_male);

      S_DB_Teachers::getInstance().add(
        { is_male,
          i1,
          i2,
          1988
        });

      ++s;
   }
}

void DB::initStudentTeacherAssocs()
{
   size_t teachers = S_DB_Teachers::getInstance().m_teacher.size();

   for( const auto & i : S_DB_Students::getInstance().m_student )
   {
      DB_Teacher::KeyId teacherId;
      teacherId.value = rand() % static_cast<int>(teachers);

      S_DBA_StudentTeachers::getInstance().add( i.getKeyId(), teacherId );
   }
}
```