```
//
// Programmer:    Craig Stuart Sapp <craig@ccrma.stanford.edu>
// Creation Date: Fri May 12 23:39:17 PDT 2006
// Last Modified: Fri Jun 23 00:12:17 PDT 2006 (subclassed to MazurkaPlugin)
// Filename:      MzSpectrogramClient.h
// URL:           http://sv.mazurka.org.uk/include/MzSpectrogramClient.h
// Documentation: http://sv.mazurka.org.uk/MzSpectrogramClient
// Syntax:        ANSI99 C++; vamp 0.9 plugin
//
// Description:   Demonstration of how to create spectral data from time data
//                supplied by the host application.
//

#ifndef _MZSPECTROGRAMCLIENT_H_INCLUDED
#define _MZSPECTROGRAMCLIENT_H_INCLUDED

#include "MazurkaPlugin.h"  // Mazurka plugin interface for Sonic Visualiser


class MzSpectrogramClient : public MazurkaPlugin {

   public:

   // plugin interface functions:

                    MzSpectrogramClient   (float samplerate);
      virtual      ~MzSpectrogramClient   ();

      // required polymorphic functions inherited from PluginBase:
      std::string   getName               (void) const;
      std::string   getMaker              (void) const;
      std::string   getCopyright          (void) const;
      std::string   getDescription        (void) const;
      int           getPluginVersion      (void) const;

      // optional parameter interface functions:
      ParameterList getParameterDescriptors (void) const;

      // required polymorphic functions inherited from Plugin:
      InputDomain   getInputDomain        (void) const;
      OutputList    getOutputDescriptors  (void) const;
      bool          initialise            (size_t channels,
                                           size_t stepsize,
                                           size_t blocksize);
      FeatureSet    process               (float **inputbufs,
                                           Vamp::RealTime timestamp);
      FeatureSet    getRemainingFeatures  (void);
      void          reset                 (void);

      // optional polymorphic functions from Plugin:
      // size_t     getPreferredStepSize  (void) const { return 0; }
      // size_t     getPreferredBlockSize (void) const { return 0; }
      // size_t     getMinChannelCount    (void) const { return 1; }
      // size_t     getMaxChannelCount    (void) const { return 1; }


   // non-interface functions and variables:

      static void   makeHannWindow        (double* output, int blocksize);
      static void   windowSignal          (double* output, double* window,
                                           float* input, int blocksize);
      static void   fft                   (int n, double *ri, double *ii,
                                           double *ro, double *io);

   private:

      double* mz_signalbuffer;  // storage space for the windowed signal
      double* mz_windbuffer;    // storage space for the signal window
      double* mz_freqbuffer;    // storage space for the complex frequency bins

      int     mz_minbin;        // minimum spectral bin to display
      int     mz_maxbin;        // maximum spectral bin to display

};


#endif // _MZSPECTROGRAMCLIENT_H_INCLUDED
```