Subject: EditMinMax ctrl implementation
Posted by pivica on Wed, 15 Feb 2006 19:52:24 GMT
View Forum Message <> Reply to Message

I have next problem. I'm using EditDouble ctrl with min and max limits. Current implementation of checking min/max boundaries isn't good for me. For example, boundary checking is done when window with EditDouble ctrl is closing. It would be much better to check limits when EditDouble ctrl lost its focus.
Well after couple of hours trying to implement some good solution in CtrlLib\EditField.cpp I give up. Problem is that I couldn't effectivly implement LostFocus() in EditValue class because EditString is inherit from it, too. In the end, here is something I've manage to implement:

```
template <class DataType, class Cv>
class EditMinMax : public EditValue<DataType, Cv> {
public:
 virtual void  LostFocus();

 // rest of class EditMinMax...
};

template <class DataType, class Cv>
void EditMinMax<DataType, Cv>::LostFocus()
{
 DataType dt;

 if (!EditMinMax<DataType, Cv>::text.IsEmpty() && IsNumber(dt)) { // Checking int and double.
  String s = EditMinMax<DataType, Cv>::text.ToString();

  CParser p(s);
  if (p.IsDouble())
   dt = p.ReadDouble(); // gcc warning: converting to `int' from `double'?
  else if (p.IsInt()) // Not doing well for doubles - returns true in 602b version?
   dt = p.ReadInt();

  if (dt < Cv::GetMin())
   s = AsString(Cv::GetMin());
  else if (dt > Cv::GetMax())
   s = AsString(Cv::GetMax());

  WString text1(s);
  EditMinMax<DataType, Cv>::text = text1;
 }

 EditField::LostFocus();
}
```

Anyway I don't like how EditMinMax ctrls are working. My opinion is that it would be much better to check limits when EditMinMax lost its focus. But if this can't be done in U++ level, I guess I will have to implement my own EditMinMax ctrl.