Subject: Re: What do you think about this approach to making CodeEditor more user extendable?
Posted by cbpporter on Thu, 17 Dec 2015 15:46:24 GMT
View Forum Message <> Reply to Message

mirek wrote on Thu, 17 December 2015 17:15OK, i have no problem with that, but perhaps it is as easy to just check current language, exactly as is happening now?

I'm doing a merge and testing stuff right now for the patch. And yes, we could probably leave things as they are in CSyntax. But some things are still a bit hard to do with a semi-hardcoded C-Like language list.

I need to add the #region and #endregion tags. So I have changed this code:

```
if(*p == '#' && findarg(highlight, HIGHLIGHT_JAVASCRIPT, HIGHLIGHT_CSS,
HIGHLIGHT_JSON) < 0) {
   static Index<String> macro;
   ONCELOCK {
    static const char *pd[] = {
     "include", "define", "error", "if", "elif", "else", "endif",
     "ifdef", "ifndef", "line", "undef", "pragma",
     // CLR
     "using"
    };
    for(int i = 0; i < __countof(pd); i++)
     macro.Add(pd[i]);
   }
```

I could add them here and that's what I'll do for the patch, but I still think it would be more elegant to have highlight be some sort of structure with fields. findarg(highlight, HIGHLIGHT_JAVASCRIPT, HIGHLIGHT_CSS, HIGHLIGHT_JSON) < 0 could become highlight.HasPreproc. And the macro list would not have to be static, but instead tied to highlight, with each separate highlight having it's own list. This way #region and #using, which are C# mostly, would not show up in other modes.