Subject: Re: Started my second attempt at redesigning CSyntax
Posted by cbpporter on Wed, 31 May 2017 12:52:58 GMT
View Forum Message <> Reply to Message

So, phase one is done.

I maintained current CodeEditor API and capabilities, but completely decoupled highlight options from language.

But beyond the current capabilities, you can also create custom languages:

```
CSyntaxOptions newLang = CSyntax::GetSyntaxDesc(CSyntax::HIGHLIGHT_CPP);
newLang.SlashBlockComments = false; // remove /* */ comments
newLang.SlashLineComment = false;   // remove // comments
newLang.PoundLineComment = true;    // the new way to comment is #
newLang.DashInId = true;            // still C++, but a-b is a valid identifier
newLang.Keywords << "event";
newLang.MacroList << "beginregion" << "endregion";

RegisterCSyntax("foo", newLang, "*.foo", "Best language");
```

Other than adding new languages like this, the main use is to allow semantic highlighting: U++ identifiers are no longer static and once per syntax. Each CodeEditor can be assigned a separate derivative syntax descriptor with a different U++ identifier list.

Now comes phase two: adding the new capabilities, in short binary literals, 100'000 numerical identifiers, #region #endregion, some extra colors and customizable macro markers/language.

Two things that I won't add for now but do kind of need are code folding and nested comment highlighting (I tried and failed).