Subject: Re: Make THISFN simpler and more powerful
Posted by Lance on Thu, 10 Oct 2024 12:06:48 GMT
View Forum Message <> Reply to Message

Lance wrote on Wed, 09 October 2024 10:17C++23 provides a remedy to write portable bitfield code, without having to resort to unnecessary bit by bit initialization/update.

Here is an example.

```
union Flags{
 int32 dummy;
 struct{
  byte borderLeft:3;
  byte borderRight:3;
  byte borderTop:3;
  byte borderBottom:3;
  byte halign:2;
  byte valign:2; //16th bit

  bool  faceNotNull:1;
  bool  boldNotNull:1;
  bool  heightNotNull:1;
  bool  widthNotNull:1;
  bool  underlineNotNull:1;
  bool  italicNotNull:1;
  bool  strikeoutNotNull:1; //23rd bit
 };

 Flags() : dummy(0){ static_assert(sizeof(*this)==sizeof(dummy)); }

 static constexpr int32 FontMask()
 {
  Flags f;
  f.faceNotNull = true;
  f.boldNotNull = true;
  f.heightNotNull = true;
  f.widthNotNull = true;
  f.underlineNotNull = true;
  f.italicNotNull = true;
  f.strikeoutNotNull = true;
  return f.dummy;
 }

 void Border(int border)
 {
  borderLeft = borderRight = borderTop = borderBottom = border;
 }
```

```
 void SetAllFontProperties()
 {
  dummy |= FontMask();
 }

 void ClearAllFontProperties()
 {
  dummy &= ~FontMask();
 }

 bool AllFontPropertiesSet()const
 {
  return ( dummy & FontMask() ) == FontMask();
 }
};
```

It's a lot more work. But there is one added benefit: if you change the bitfields definition and add a few bits in front of the font properties group, you have a better chance not to break existing code unknowingly.

with std=c++23, gcc and msvc accept the above code. But clang rejects it, for good reason. The union default constructor used in constexpr ...FontMask()const should be constexpr modified too.

```
 constexpr Flags() : dummy(0){
  static_assert(sizeof(*this)==sizeof(dummy));
      }
```