Overview

CryFont is used to generate font textures that are required to render text on the screen. The various features of the font rendering can be seen by using the **r_DebugFontRendering** console variable.

The output is not only to test the functionality but also to document how the features can be used.

Supported Features

CryFont supports the following features:

- Font shaders to configure the appearance of fonts. Multiple passes with configurable offset and color are supported to enable generation of shadows or outlines.

  The following **XML** file shows a sample font shader:

```
<fontshader>
  <font path="VeraMono.ttf" w="288" h="416"/>
  <effect name="default">
    <pass>
      <color r="0" g="0" b="0" a="1"/>
      <pos x="1" y="1"/>
    </pass>
  </effect>
  <effect name="console">
    <pass>
      <color r="0" g="0" b="0" a="0.5"/>
      <pos x="2" y="2"/>
    </pass>
  </effect>
</fontshader>
```

The attributes **w** and **h** of the XML font element specify the width and height of the font texture. The order of the passes in XML defines the order in which the passes are rendered. A **pass** element without child elements means that the pass is rendered with the default settings. The tag **pos** allows to offset the font, while **color** is used to tint it and define the transparency with the alpha channel.

- Unicode.
  Note: The default font we use deliberately does not support all unicode characters to save memory but other fonts can be used.
- Truetype fonts as source, cached in a small texture (common characters are precached but runtime updates are possible and supported).
- Colored text rendering.
- Adjustable transparency.
- Varying colors within a string.
  Usage: $ 0..9 to set one of the 10 available colors, $$ to print the $ symbol and $o to switch off the feature.
- Returns and tabs within a string.
- Alignment (left, center and right).
- Computation of the width and height of a string (used internally for center and right alignment).
- Different font sizes.
  Note: Bilinear filtering allows some blurring but as no mip-maps are used, this has limitations in minification.
- Proportional and monospace fonts.
- Pixel perfect rendering with exact texel to pixel mapping for best quality.