Client-side function

DxDrawText

Draws a string of text on the screen for one frame. In order for the text to stay visible continuously, you need to call this function with the same parameters on each frame update (see onClientRender).

Syntax

```
bool dxDrawText ( string text, float leftX, float topY, [ float rightX = leftX, float bottomY = topY, int color = white, floa
t scaleXY = 1.0, float scaleY = 1.0,
                  mixed font = "default", string alignX = "left", string alignY = "top", bool clip = false, bool wordBreak =
false,
                  bool postGUI = false, bool colorCoded = false, bool subPixelPositioning = false,
                  float fRotation = 0.0, float fRotationCenterX = 0.0, float fRotationCenterY = 0.0, float fLineSpacing = 0.0
 ] )
```

Required Arguments

- **text:** the text to draw
- **leftX:** the absolute X coordinate of the top left corner of the text
- **topY:** the absolute Y coordinate of the top left corner of the text

Optional Arguments

- **rightX:** the absolute X coordinate of the right side of the text bounding box. Used for text aligning, clipping and word breaking.
- **bottomY:** the absolute Y coordinate of the bottom side of the text bounding box. Used for text aligning, clipping and word breaking.
- **color:** the color of the text, a value produced by tocolor or 0xAARRGGBB (AA = alpha, RR = red, GG = green, BB = blue).
- **scale:** the size of the text.**scale:** can (optionally) be specified as two floats. i.e. **scaleX, scaleY**
- **font:** Either a custom DX font element or the name of a built-in DX font: **Note: Some fonts are incompatible with certain languages such as Arabic.**
  - **"default":** Tahoma
  - **"default-bold":** Tahoma Bold
  - **"clear":** Verdana
  - **"arial":** Arial
  - **"sans":** Microsoft Sans Serif
  - **"pricedown":** Pricedown (GTA's theme text)
  - **"bankgothic":** Bank Gothic Medium
  - **"diploma":** Diploma Regular
  - **"beckett":** Beckett Regular
  - **"unifont":** Unifont
- **alignX:** horizontal alignment of the text within the bounding box. Can be **"left"**, **"center"** or **"right"**.
- **alignY:** vertical alignment of the text within the bounding box. Can be **"top"**, **"center"** or **"bottom"**.
- **clip:** if set to *true*, the parts of the text that don't fit within the bounding box will be cut off.
- **wordBreak:** if set to *true*, the text will wrap to a new line whenever it reaches the right side of the bounding box. If *false*, the text will always be completely on one line.
- **postGUI:** A bool representing whether the text should be drawn on top of or behind any ingame GUI (rendered by CEGUI).

- **colorCoded:** Set to true to enable embedded #FFFFFF color codes. **Note: clip and wordBreak are forced false if this is set.**
- **subPixelPositioning:** A bool representing whether the text can be positioned sub-pixel-ly. Looks nicer for moving/scaling animations.

- **fRotation:** Rotation
- **fRotationCenterX:** Rotation Origin X
- **fRotationCenterY:** Rotation Origin Y

- **fLineSpacing:** Distance in pixels between the lines of text, this can be a negative number, works only when **colorCoded** is set to true

Returns

Returns *true* if successful, *false* otherwise.

Remarks

The function is known to *optimize* certain drawing scenarios related to scaling and opacity (so called **text on raster**

optimisation). You can find out more about it here.