



# USDText

Proposal for representing text in USD

Autodesk



# *“We make software for people who make things”*

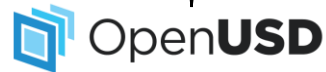
Architecture, Engineering,  
and Construction



Product Design and  
Manufacturing



Media and Entertainment



**Extensive use of workflows that require transcription and transfer of design data.**

# Text in Architecture, Engineering, and Construction

**TABLE 1: ELECTRICAL LOADS**

Item	Value
TOTAL COMPUTED LOAD	176,258 VA
TOTAL COMPUTED AMPS	4,536 AMPS
RECOMMENDED SERVICE @ 120/208V, 3ø, 4W	4,500 AMPS

**TABLE 2: RECOMMENDED SERVICE**

Item	Value
RECOMMENDED SERVICE @ 120/208V, 3ø, 4W	4,500 AMPS

**TABLE 3: TOTAL COMPUTED LOAD "BUILDING A AND B"**

Item	Value
RECOMMENDED SERVICE @ 120/208V, 3ø, 4W	4,500 AMPS

**REFERENCE NOTES**

- 1) INCOMING SERVICE A FEEDER CABLES AND CONDUIT PER LADWP REQUIREMENTS.
- 2) 1" C. 1 $\frac{1}{2}$ "/0 CU. GROUND TO COLD WATER PIPE, CONNECT WITHIN 5 FT. OF BUILDING ENTRY.
- 3) 1" C. 1 $\frac{1}{2}$ "/0 CU. GROUND BOND TO BUILDING STEEL AND CONCRETE FOUNDATION FOOTING REBAR.
- 4) 3/0 BARE CU. GROUND BONDED TO (3) 10' X 3/4" COPPER CLAD GROUND RODS LOCATED BELOW B2 GARAGE PARKING LEVEL (BELOW METHANE BARRIER), CONNECTED IN DELTA CONFIGURATION, MINIMUM 10' APART. PROVIDE EXOTHERMIC CONNECTIONS TO GROUND RODS. WHERE BARE CU PENETRATE THROUGH FOUNDATION PROVIDE 12" CONCRETE DAM.
- 5) MAIN GROUNDING BUS ASSEMBLY, TO BE LOCATED WITHIN MAIN ELECTRICAL ROOM.
- 6) 1" C - 1 $\frac{1}{2}$ "/0 THIN CU. (GROUND)
- 7) PROVIDE BARE 3/0 CU. BONDING JUMPER BETWEEN BUS BAR AND SWITCHGEAR FRAME
- 8) SHUNT TRIP SHALL BE INTERCONNECT WITH FA TO COMPLY WITH ELEVATOR ANSI STANDARD

Credit: Breen Design Group  
Electrical Diagram authored in AutoCAD

# Text in Architecture, Engineering, and Construction

The screenshot shows the Revit software interface. On the left, a drawing of a closet is visible with dimensions and annotations. A 'Type Properties' dialog box is open in the center, showing the following settings:

- Family: System Family: Text
- Type: 3/32" text
- Graphics:
  - Color: Black
  - Line Weight: 1
  - Background: Transparent
  - Show Border:
  - Leader/Border Offset: 5/64"
  - Leader Arrowhead: 15 Degree Filled Arrow
- Text:
  - Text Font: Arial
  - Text Size: 3/32"
  - Tab Size: 1/2"
  - Bold:
  - Italic:
  - Underline:
  - Width Factor: 1.000000

Annotations in the drawing include: 'CLOSET SEE ENLARGED PLAN', 'SUPPLY AIR SHAFT FOR CORRIDOR, SEE MECH. DWGS FOR SIZE', 'SEE SHT A9 05 FOR WALL TYPES DETAILS', 'PROVIDE FIRE DAMPER FOR RATED WALL PENETRATION, SEE MECH. DWGS.', and 'CONC. BEAM ABOVE & BELOW. SEE PLAN'.

This image displays a grid of architectural details from a Revit project. The details are organized into a grid with labels and sheet numbers. The details include:

- BATHUB DETAIL (NR) 20
- LOW LEVEL EXIT SIGN 14
- SUSPENDED ACOUSTICAL TILE CEILING 6
- F.E. CABINET - 1 hr 25
- F.E. CABINET - 2 hr 24
- BATHUB DTL (1 - HR) 19
- WALL GUARD 13
- SUSPENDED GYPSUM BOARD CEILING 5
- FAN COIL UNIT 23
- BATHUB DTL (2-HR) 18
- MULLIONS DETAIL 12
- ACCESS PANEL DETAIL 4
- SUPPLY AIR SHAFT 22
- F.C.U. PLANG/GUESTRM 17
- TERRAZZO JOINT 11
- GUESTRM. WALL DET 16
- HORIZ. FAN COIL UNIT 10
- WALL HEAD DETAILS 8
- DTL @ METAL DECK 2
- PLUMB EXHAUST SHAFT 21
- GUESTRM. WALL DET 15
- CLG. EXPANSION JNT 9
- T-BAR CEILING 7
- BRACED WALL DETAIL 1

On the right side of the grid, there is a table with the following columns: 'No.', 'Description', and 'Date'. Below the table, the project name 'Autodesk Revit Hotel 5' and 'INTERIOR DETAILS' are listed. At the bottom right, the sheet number 'A9.60' is displayed.

Engineering and Construction Diagram using Revit



# Text in Product Design and Manufacturing

The image displays the Autodesk Fusion 360 interface. On the left, the 'BROWSER' panel shows a tree view with 'Sketch' and 'Text' highlighted. A 'Text' command panel is open, showing options for font, height, color, and justification. A tooltip for the 'Text' command is visible, explaining that it creates text inside a rectangular frame or along a selected path. The main workspace shows a technical drawing of a steamer with four numbered callouts (1, 2, 3, 4) pointing to different parts of the assembly. A text box in the drawing contains the following notes:

NOTES: UOS  
1. ALL UNITS MM  
2. REFER TO BOM 3394-B

On the right side of the drawing, there is a table with the following structure:

Dept.	Technical reference	Created by
		Mohammed Ali
		Document type
		Title
		Steamer desi

The bottom of the interface shows the 'COMMENTS' panel and a row of icons for various text-related commands.

Annotations in Fusion 360 Design

# Text in Product Design and Manufacturing

The screenshot displays the Autodesk Fusion 360 Electronics workspace. The top toolbar includes tabs for DESIGN, DOCUMENT, VALIDATE, AUTOMATE, and LIBRARY, along with various tool icons. The main workspace shows a circuit schematic for an ATTINY84 microcontroller (U1). The microcontroller is connected to a power supply (VCC) and ground (GND) through a capacitor (C1). A switch (SW1) is connected to pin 14 (GND) and pin 1 (VCC). A 6-pin connector (J1) is connected to pins 4 (RST), 5 (MISO), 6 (MOSI), 7 (SCLK), and 8 (GND). A 5-pin connector (J2) is connected to pins 9 (MISO), 10 (SCLK), 11 (RST), 12 (MISO), and 13 (RST). The schematic also shows four LEDs (LED1-LED4) connected to pins 1, 2, 3, and 4 through resistors (R1-R4). The left sidebar shows a layer set of 'All Layers' and a list of layers including SimResults, SimProbes, Modules, Nets, Busses, Pins, Symbols, Names, Values, Info, Guide, and SpiceOrder. The bottom status bar indicates 'Left-click diagonal drag to select objects'.

Autodesk Fusion

DESIGN DOCUMENT VALIDATE AUTOMATE LIBRARY

SWITCH VIEW EDIT PLACE CONNECT SIMULATE REWORK MODIFY SHORTCUTS SELECT

DISPLAY LAYERS 91 Nets 0.1 inch (3.8 6.5) Click or press / to activate command line mode

Layer Set All Layers

Your design uses managed libraries that have not yet been downloaded. [Open Managed Libraries to download and use these libraries.](#)

Layer names are updated for all new files. Legacy files retain the former naming convention. [Click here](#) for more info.  Don't show again

88 SimResults

89 SimProbes

90 Modules

91 Nets

92 Busses

93 Pins

94 Symbols

95 Names

96 Values

97 Info

98 Guide

99 SpiceOrder

Details

ERRORS Left-click diagonal drag to select objects

SHEETS

INSPECTOR SELECTION FILTER

VCC

6 - D4

7 - D5

8 - D6

9 - D7

10 - D8

J1

1 2

3 4

5 6

VCC

MISO

SCLK

MOSI

GND

J2

1 2

3 4

5 6

MISO

SCLK

RST

GND

U1

VCC (PCINT11/-RESET/DW/PB3

(PCINT10/INT0/OC0A/CKOUT)/PB2

(PCINT9/XTAL2/PB1

(PCINT8/XTAL1/CLK1)/PB0

(PCINT7/CP/OC0B/ADC7)/PA7

(PCINT6/OC1A/SB0/MOSI/ADC6)/PA6

(PCINT5/OC1B/MISO/DO/ADC5)/PA5

(PCINT4/T1/SCL/USCK/ADC4)/PA4

(PCINT3/T0/ADC3)/PA3

(PCINT2/AIN1/ADC2)/PA2

(PCINT1/AIN0/ADC1)/PA1

(PCINT0/AREF/ADC0)/PA0

ATTINY84

GND

C1

SW1

D1

P1

LED1

LED2

LED3

LED4

R1

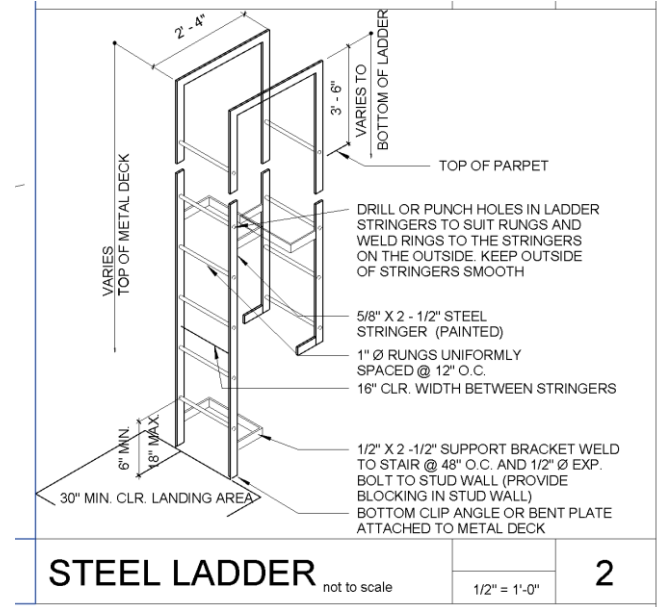
R2

R3

R4

SW2

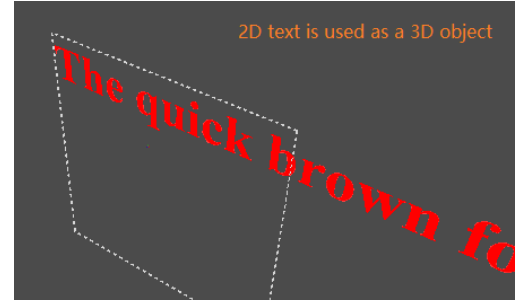
# Text in Product Design and Manufacturing





# UsdText: Text Primitive in USD

- 2D text primitive in object space.
  - Single or Multiple lines.
  - 2D in screen space, such as UI and annotations.
  - 3D object in the world.
- Common face styles and layout.
  - Artistic styles are not generally used architectural and manufacturing design workflows.



# Attributes of a text primitive

- Encoding (USD supports UTF-8)
- Script:

## Latin

### Typography

[Article](#) [Talk](#)

From Wikipedia, the free encyclopedia

## Han

### 字体排印学 [\[编辑\]](#)

[条目](#) [讨论](#) [汉](#) [漢](#) [大陆简体](#) [▼](#)

维基百科，自由的百科全书

## Devanagari

### अक्षर कला [◄](#)

[लेख](#) [संवाद](#)

मुक्त ज्ञानकोश विकिपीडिया से

## Arabic

### طباعة المحارف المنضدة

[مقالة](#) [نقاش](#)

- Typeface (font family):
  - Font name: Arial, Consolas, Times New Roman, etc.
  - Font Style: Regular, **Bold**, *Italic*, ***Bold Italic***.

# Attributes of a text primitive

- Spacing styles:

- Weight

Weight = 200

The quick brown fox

Weight = 400

The quick brown fox

- Height

Height = 11

The quick brown fox

Height = 9

The quick brown fox

- Width

Width factor is 100%

The quick brown fox

Width factor is 150%

**The quick brown fox**

- Oblique

*The quick brown fox*

- Different from italic. The angle can be user defined.

- Character space

Normal character space

The quick brown fox

Character space is expanded by 1.6

The quick brown fox

# Attributes of a text primitive

- Styles for emphasis:

- Underline      The quick brown fox

- Overline      The quick brown fox

- Strikethrough      ~~The quick brown fox~~

- Text Direction

- Most of the western scripts are from left to right.
- Some scripts are from right to left: Arabic, Hebrew and so on.
- Chinese can be written from left to right, right to left or top to bottom.

Left to right      Top to bottom

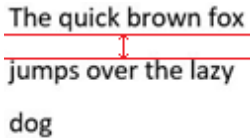
龍年大吉      龍  
                                 年  
Right to left      吉  
                                 大  
吉大年龍      吉

# Attributes of Multiline text

- Paragraph Style (for multiline text)

- Line Space

Line space

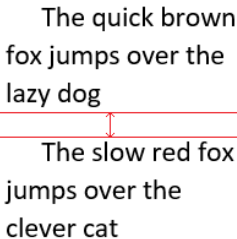


The quick brown fox  
jumps over the lazy  
dog

The diagram shows two lines of text: "The quick brown fox" and "jumps over the lazy dog". A vertical double-headed arrow is positioned between the two lines, indicating the vertical distance between them, which is labeled "Line space".

- Paragraph Space

Paragraph space

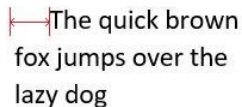


The quick brown  
fox jumps over the  
lazy dog  
The slow red fox  
jumps over the  
clever cat

The diagram shows two paragraphs. The first paragraph is "The quick brown fox jumps over the lazy dog" and the second is "The slow red fox jumps over the clever cat". A vertical double-headed arrow is positioned between the two paragraphs, indicating the vertical distance between them, which is labeled "Paragraph space".

- Indent

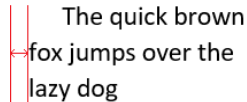
First line indent



The quick brown  
fox jumps over the  
lazy dog

The diagram shows the text "The quick brown fox jumps over the lazy dog" where the first line is indented to the right. A vertical double-headed arrow is positioned between the start of the first line and the start of the second line, indicating the indentation, which is labeled "First line indent".

Left indent

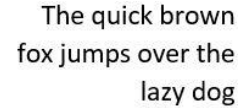


The quick brown  
fox jumps over the  
lazy dog

The diagram shows the text "The quick brown fox jumps over the lazy dog" where all lines are indented to the right. A vertical double-headed arrow is positioned between the start of the first line and the start of the second line, indicating the indentation, which is labeled "Left indent".

- Alignment

Right alignment



The quick brown  
fox jumps over the  
lazy dog

The diagram shows the text "The quick brown fox jumps over the lazy dog" where all lines are aligned to the right. A vertical double-headed arrow is positioned between the start of the first line and the start of the second line, indicating the alignment, which is labeled "Right alignment".

- Tab stops

Left tab stop

Name	Age
Felix	18
Rosie	16
Alexander	18

The diagram shows a table with two columns: "Name" and "Age". The text is aligned to the left of each column, which is labeled "Left tab stop".

# Attributes of Multiline text

- Column Style (for multiline text)

- Lines direction
  - Special for top-to-bottom Chinese

Right to left

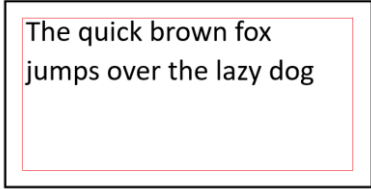
上海市是中国第一大城市。它位于中国东部，属华东地区。

Left to right

部，属华东地区。上海市是中国第一大城市。它位于中国东部。

- Margins

Four margins

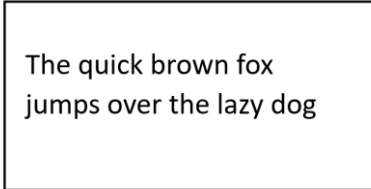


The quick brown fox  
jumps over the lazy dog

A diagram illustrating text with four margins. The text "The quick brown fox jumps over the lazy dog" is centered within a rectangular box. A smaller, inner rectangular box is drawn around the text, with a thin red border, representing the margins. The text is aligned to the top-left of this inner box.

- Vertical alignment

Center alignment



The quick brown fox  
jumps over the lazy dog

A diagram illustrating text with center alignment. The text "The quick brown fox jumps over the lazy dog" is centered within a rectangular box. The text is aligned to the top-left of the box, but its horizontal position is centered relative to the box's width.

# Other considerations for Text

- Font substitution
  - Choose another font if the current font can not support the character.

The dragon year is called 龙年 in Chinese.

The font is Times New Roman.

Change to  
Dengxian  
for Chinese  
characters.

The following characters  
still use Times New Roman.

- Complex scripts



Exmaple from [Complex text layout – Wikipedia](#)  
To illustrate the complex ligatures of devanagari

# Other considerations for Text

- Markup formats
  - A multiple line text with complex layout and styles always use text string with markups.
  - Common markup formats:
    - Rich Text Format
    - HTML
  - Internal markup format.
- Unit of the font metrics
  - The same as world unit if it is a 3D object.
  - If it is 2D in screen space, the unit could be pixel, or publishing point.
- Text in a path.
  - Not common in architectural and manufacturing design but could be an extension.



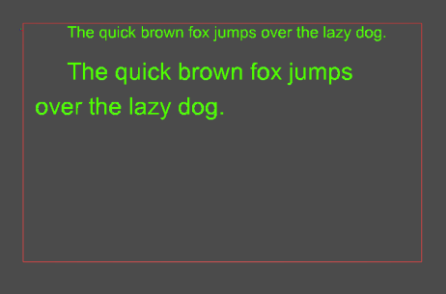
# USD-Text proposal

<https://github.com/autodesk-forks/USD-proposals/tree/adsk/feature/text/proposals/text>

- **SimpleText** (IsA schema /Gprim)  
Defines a *single line* single style text prim
  - API schema:
    - TextStyle
    - TextLayout
  - Hydra Prims:
    - HdStSimpleText
- **MarkupText** (IsA schema /Gprim)  
Defines a *multiline* multiple style text prim
  - API schema:
    - ColumnStyle
    - ParagraphStyle
  - Hydra Prims:
    - HdStMarkupText



The quick brown fox



The quick brown fox jumps over the lazy dog.  
The quick brown fox jumps  
over the lazy dog.

*Please refer to proposal for complete set of schema properties*

# Example: SimpleText

```
def SimpleText "Text" (){
  uniform string textData = "The quick brown fox"
  color3f[] primvars:displayColor = [(1, 1, 0)]
  rel textStyle:binding = </Style>
  rel material:binding = </TextRenderer>
  uniform string renderer = "TextRenderer"
}

def TextStyle "Style" {
  uniform string typeface = "Times New Roman"
  uniform int textHeight = 100
  uniform bool bold = 1
  uniform string overlineType = "normal"
}

# Bind to a material
def Material "TextRenderer" {
  token outputs:surface.connect =
    </TextRenderer/TextShader.outputs:surface>

  def Shader "TextShader" {
    uniform token info:id = "TextRendererSurface"
    token outputs:surface
  }
}
```



The quick brown fox

# Example: MarkupText

```
def MarkupText "TextA" (  
) {  
    uniform string markupString =  
    "{\rtf1\fbidis\ansi\ansicpg1252\deff0\deflang1033{\fonttbl{\f0\fswiss\frq2  
\fcharset0 Calibri;}{\f1\fmmodern\frq1\fcharset0 Consolas;}}{\colortbl  
;\red255\green0\blue0;}\viewkind4\uc1\pard\ltrpar\s160\s1252\slmult1\kerni  
ng2\f0\fs22 The quick brown \f1 fox \line\pard\ltrpar jumps over \cf1\u1  
the lazy dog.\cf0\kerning0\ulnone\par}"  
    uniform token markupLanguage = "rtf"  
  
    color3f[] primvars:displayColor = [(0, 0, 0)]  
    rel textStyle:binding = </Style>  
    rel columnStyle:binding = </column>  
    rel paragraphStyle:binding = </paragraph>  
    rel material:binding = </TextRenderer>  
    uniform string renderer = "TextRenderer"  
}  
  
def TextStyle "Style" {  
    uniform string typeface = "Times New Roman"  
    uniform int textHeight = 11  
}  
  
def ColumnStyle "column" {  
    uniform float columnWidth = 500  
    uniform float columnHeight = 300  
    uniform float2 offset = (0.0, 0.0)  
}
```

```
def ParagraphStyle "paragraph" {  
    uniform float leftIndent = 15.0  
    uniform float rightIndent = 30.0  
    uniform float firstLineIndent = 0.0  
    uniform float paragraphSpace = 15.0  
}  
  
def Material "TextRenderer"  
{  
    token outputs:surface.connect =  
</TextRenderer/TextShader.outputs:surface>  
  
    def Shader "TextShader"  
    {  
        uniform token info:id = "TextRendererSurface"  
        token outputs:surface  
    }  
}
```

The quick brown fox

jumps over the lazy dog.

# UsdText API Plugins

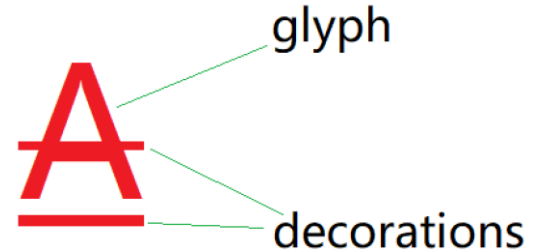
## API classes to generate visual representation for Text

- **UsdImagingRenderer:**
  - Generates the geometry and textures for each character in text primitive
- **UsdImagingText:**
  - Consumes the attributes from a text primitive.
  - Handles the Typeface properties e.g. (metrics or control points of the outline) from the font file
  - Uses UsdImagingRenderer to translate them into renderable items.
- **UsdImagingMarkupParser:**
  - Parses markup data and generates structure of text.
  - Can be extended to create custom parsers.  
e.g., a RTFParser plugin, which inherits from UsdImagingMarkupParser.

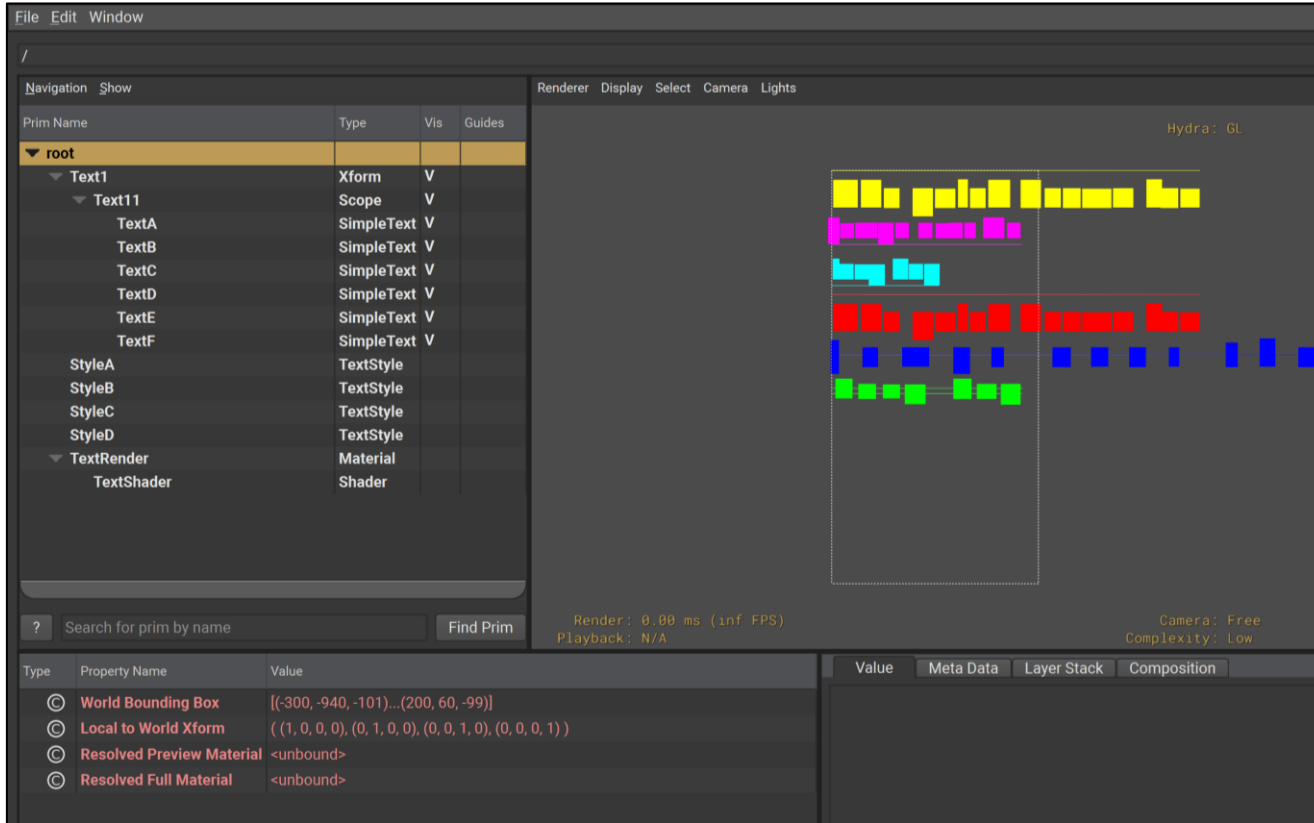
# Text prims in Hydra

HdStSimpleText and HdStMarkupText

- A character is composed from a *glyph* and *decorations* (underline, overline and strikethrough).
- Hydra prims will generate one draw item for each of the glyph for all the characters. These are then consolidated.
- The draw items are rendered as textured quad that refers to an texture atlas for every character.
- The decorations are rendered as separate draw items by decomposing them to basisCurves rprims to render them as lines.
- Storm implementation:
  - New shader for text (text.glsfx)

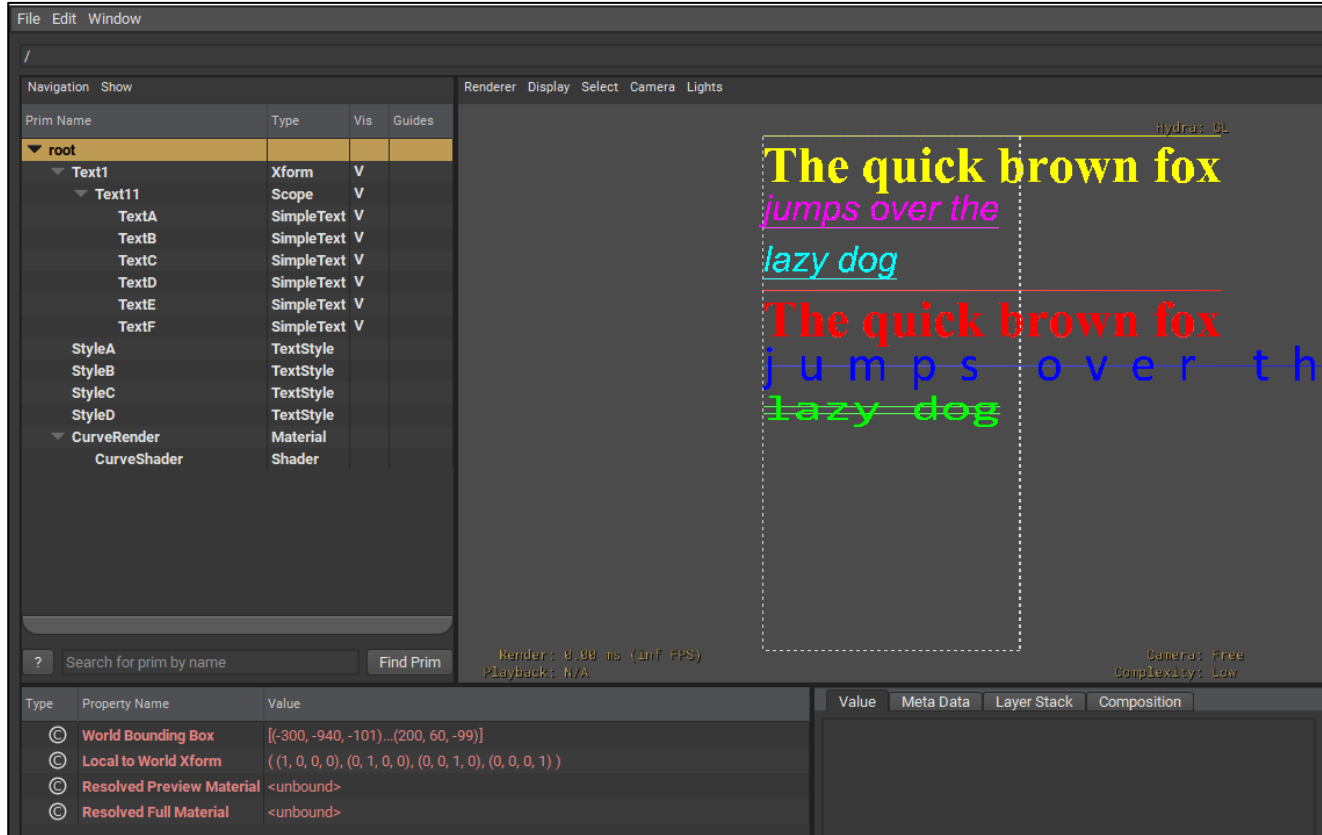


# Example: UsdText in USDView (Storm)



Using a simpler implementation of UsdImagingRenderer that demonstrates UsdImagingText capabilities

# Example: UsdText in USDView (Storm)



Using a font elaborator implementation UsdImagingRenderer that demonstrates UsdImagingText

# Next steps

## Call for community participation

- Provide feedback on UsdText proposal.
- Seeking partners to review implementation and help accelerate the proposal.
- Prototype UsdText schema with your renderers.
- Proposal: <https://github.com/autodesk-forks/USD-proposals/tree/adsk/feature/text/proposals/text>
- Implementation: Will be published soon to <https://github.com/autodesk-forks/USD>





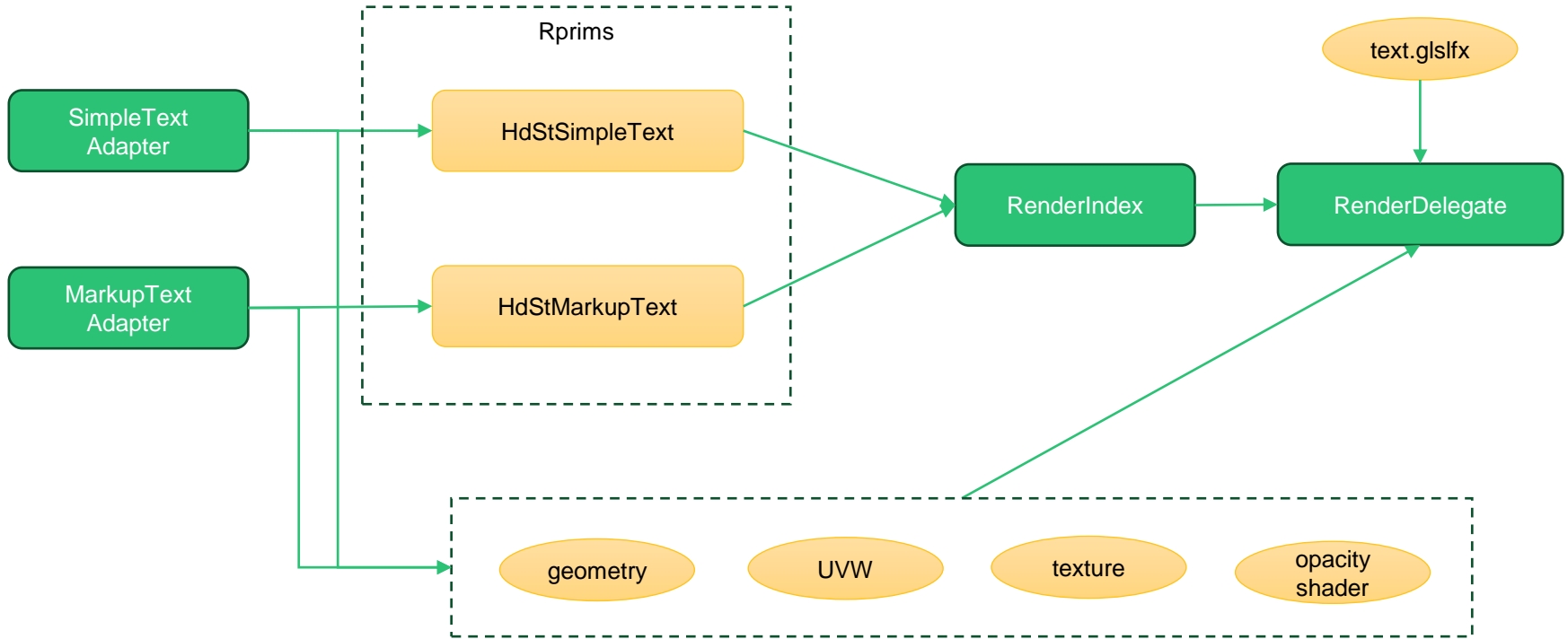
Autodesk and the Autodesk logo are registered trademarks or trademarks of Autodesk, Inc., and/or its subsidiaries and/or affiliates in the USA and/or other countries. All other brand names, product names, or trademarks belong to their respective holders. Autodesk reserves the right to alter product offerings, specifications and pricing at any time without notice, and is not responsible for typographical or graphical errors that may appear in this document.

© 2024 Autodesk. All rights reserved.

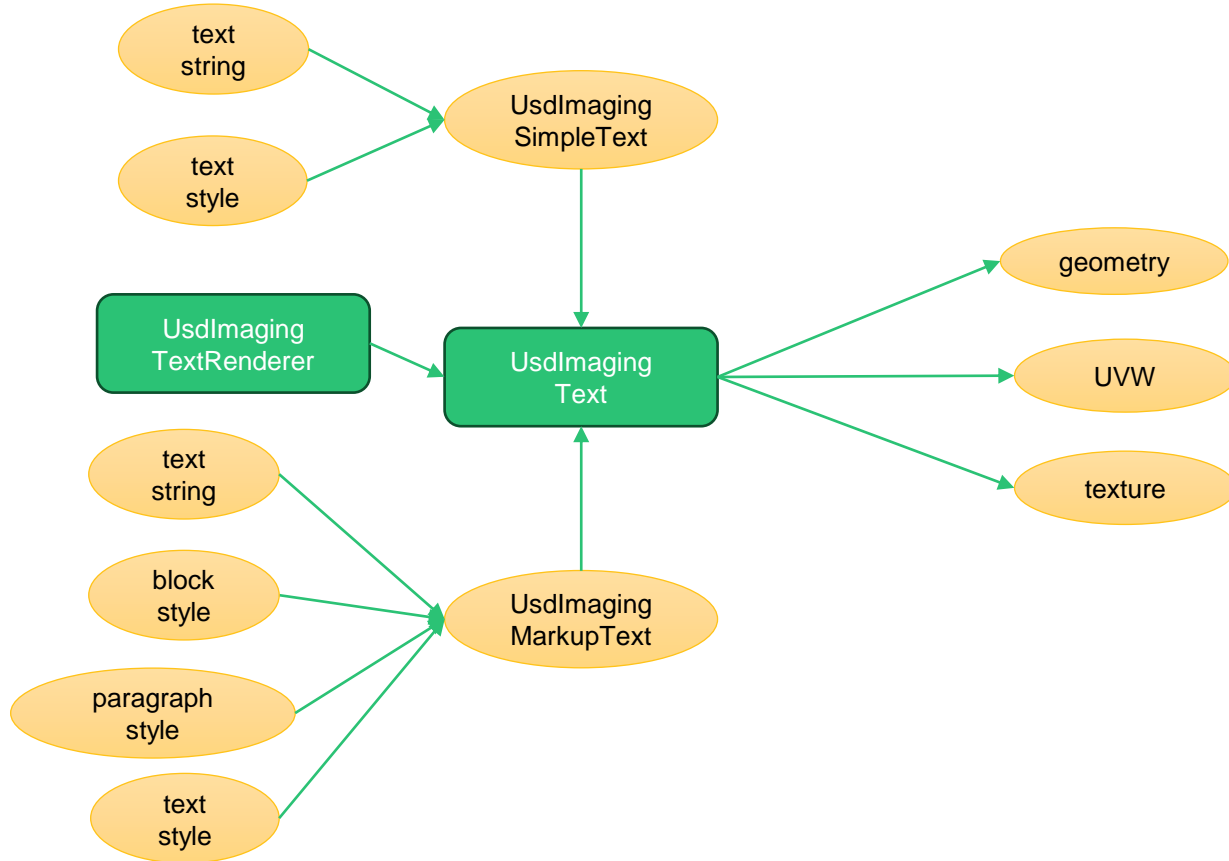


# Supplemental

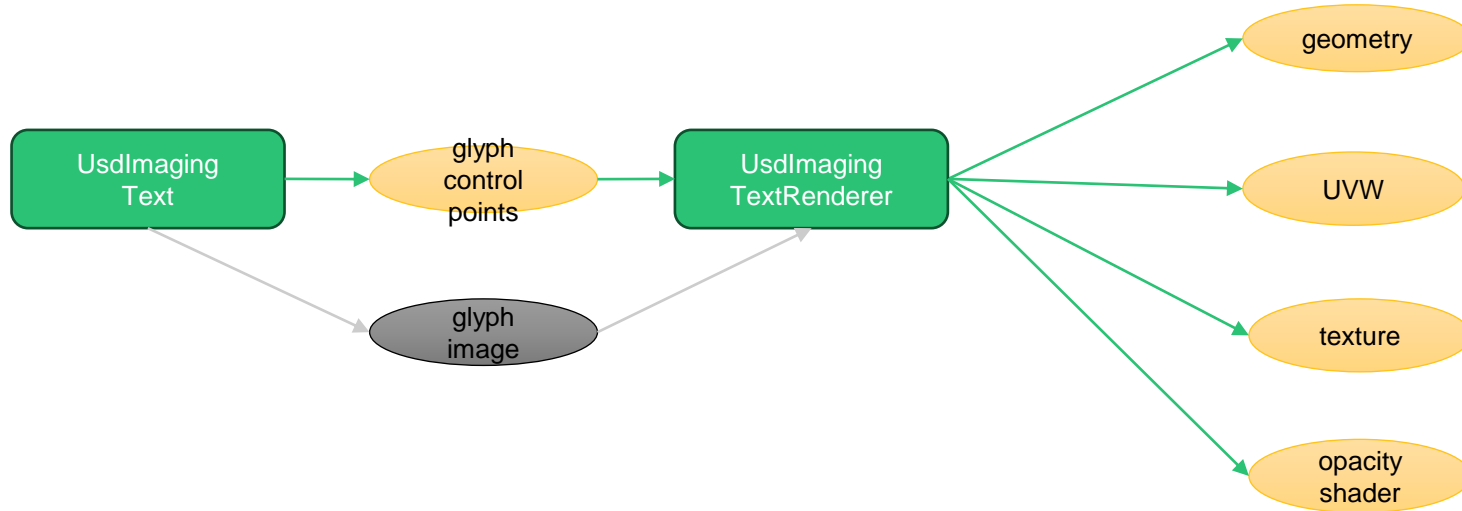
# Current implementation



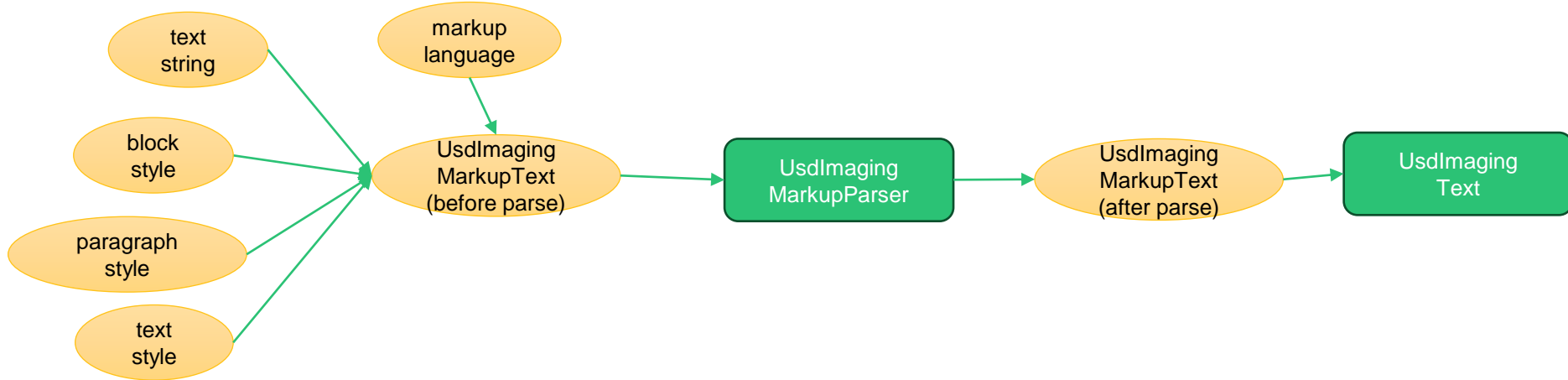
# The UsdImagingText plugin



# The UsdImagingTextRenderer plugin



# The UsdImagingMarkupParser plugin



# The text.glslfx

- The VS shader is simple: compute the position using the matrix, and send the textColor, textOpacity and UVWs to FS.
- The FS shader:

```
void main(void)
{
    float alpha = inData.TextOpacity;
    alpha = alpha * getOpacity(inData.UVW);

    // The curve primitive have alpha natively. So here we first get the override color, then
    // multiply the alpha of the primitive with the override alpha, and finally set the alpha
    // to the final color.
    vec4 overrideColor = ApplyColorOverrides(vec4(inData.TextColor, 1.0));
    alpha = alpha * overrideColor.a;
    vec4 finalColor = vec4(overrideColor.rgb, alpha);
    vec3 Peye = inData.Peye.xyz / inData.Peye.w;
    RenderOutput(vec4(Peye, 1), vec3(0, 0, 1), finalColor, vec4(1));
}
```