Graphics Programmer's Guide

Version 0.33.0 (Beta)

Component Layouts and Templating

Template system general overview

Template input data is used within the graphics template editor to populate dynamic content using a templating language called Handlebars.

Template input data

Each graphic, depending on the category of meeting data that it consumes and the data types contained within each category, will have distinct visual elements that will represent each of those data types. The following is a list of the meeting data categories that the Zoom Graphics Toolkit currently supports and the graphic types that consume them:

- **Meeting Chat Data**: Chat Scroll Graphic, Chat Highlight Graphic
- **Meeting QA Data**: QA Scroll Graphic, QA Highlight Graphic
- **Meeting Poll Data**: Poll Result Graphic
- **Meeting Emoji Reaction Data**: Emoji Reaction Graphic
- **Meeting Caption Data**: Closed Caption Graphic

As stated above, the Chat Scroll Graphic consumes the Meeting Chat Data category. Composed in the meeting chat data is a list of **`` `chat_message` ``** and **`` `chat_reply` ``** objects. These are similar but distinct data types with different attributes. A `` `chat_reply` `` object in this case has all the properties of a `` `chat_message` `` object and an additional property called "parentMessage". The value of the "parentMessage" property is a reference to the chat message that started the thread that the chat reply belongs to.

All meeting data are JSON-serializable JavaScript objects under the hood. Therefore, we can define the data types of each graphic type more formally using the TypeScript syntax. The type definitions for each graphic type are as follows:

- Chat scroll graphic template input
- Chat highlight graphic template input
- Closed captions graphic
- Emoji reaction graphic
- Poll result graphic
- QA highlight graphic
- QA scroll graphic

Chat scroll graphic template input

chat_message

```
type ChatMessageData = {
  messageID: string
  threadID: string
  timestamp: number
  chatMessageType: number
  senderUserID: string
  senderDisplayName: string
  senderAvatarPath: string | undefined
  msgContent: string
  isComment: boolean
  isChatToWaitingRoom: boolean
  isThread: boolean
  isEmphasized: boolean
  textStyleItemList: any[]
}
```

chat_reply

```
type ChatReplyData = {
  messageID: string
  threadID: string
  timestamp: number
  chatMessageType: number
  senderUserID: string
  senderDisplayName: string
  senderAvatarPath: string | undefined
  msgContent: string
  isComment: boolean
  isChatToWaitingRoom: boolean
  isThread: boolean
  isEmphasized: boolean
  textStyleItemList: any[]
  parentMessage:
    | {
        messageID: string
        threadID: string
        timestamp: number
        chatMessageType: number
        senderUserID: string
        senderDisplayName: string
        senderAvatarPath?: string
        msgContent: string
        isComment: boolean
        isChatToWaitingRoom: boolean
        isThread: boolean
        isEmphasized: boolean
        textStyleItemList: any[]
      }
    | undefined
}
```

Chat highlight graphic template input

chat_message

```
type ChatMessageData = {
  messageID: string
  threadID: string
  timestamp: number
  chatMessageType: number
  senderUserID: string
  senderDisplayName: string
  senderAvatarPath: string | undefined
  msgContent: string
  isComment: boolean
  isChatToWaitingRoom: boolean
  isThread: boolean
  isEmphasized: boolean
  textStyleItemList: any[]
}
```

chat_reply

```
type ChatReplyData = {
  messageID: string
  threadID: string
  timestamp: number
  chatMessageType: number
  senderUserID: string
  senderDisplayName: string
  senderAvatarPath: string | undefined
  msgContent: string
  isComment: boolean
  isChatToWaitingRoom: boolean
  isThread: boolean
  isEmphasized: boolean
  textStyleItemList: any[]
  parentMessage:
    | {
        messageID: string
        threadID: string
        timestamp: number
        chatMessageType: number
        senderUserID: string
        senderDisplayName: string
        senderAvatarPath: string | undefined
        msgContent: string
        isComment: boolean
        isChatToWaitingRoom: boolean
        isThread: boolean
        isEmphasized: boolean
        textStyleItemList: any[]
      }
    | undefined
}
```

Closed captions graphic

single_language_caption

```
type singleLanguageCaptions = {
  messageID: string
  speakerID: number
  speakerName: string
  messageContent: string
  timeStamp: number
  messageType: number
  senderAvatarPath?: string
  isOriginal: boolean
  isSim?: boolean
  strMarker?: number
}[]
```

multi_language_caption

```
type multiLanguageCaptions = {
  original: {
    messageID: string
    speakerID: number
    speakerName: string
    messageContent: string
    timeStamp: number
    messageType: number
    senderAvatarPath?: string
    isOriginal: boolean
    isSim?: boolean
    strMarker?: number
  }
  translation?: {
    messageID: string
    speakerID: number
    speakerName: string
    messageContent: string
    timeStamp: number
    messageType: number
    senderAvatarPath?: string
    isOriginal: boolean
    isSim?: boolean
    strMarker?: number
  }
}[]
```

traditional_captions

```
type traditionalCaptions = {
  messageID: string
  speakerID: number
  speakerName: string
  messageContent: string
  timeStamp: number
  messageType: number
  senderAvatarPath?: string
  isOriginal: boolean
  isSim?: boolean
  strMarker?: number
}
```

Emoji reaction graphic

The emoji graphic currently does not expose any meeting data to the templates.

Poll result graphic

question_title

```
type PollQuestionItemData = {
  pollingID: string
  pollingQuestionID: string
  pollingQuestionName: string
  pollingQuestionType: 1 /* SingleSelect */ | 2 /* MultiSelect */
  answeredCount: number
  isRequired: boolean
  pollingSubQuestionItemList: {
    pollingParentQuestionID: string
    pollingSubQuestionID: string
    pollingQuestionName: string
  }[]
  questionImagePath: string | undefined
  visMethod: 'pie-chart' | 'bar-chart'
  graphicStatus: { showingResults: boolean; showingQuestion: boolean }
  result: {
    totalResponseCount: number | undefined
    questionResults:
      | {
          [answerID: string]: {
            answerID: string
            answerValue: string | undefined
            frequency: number
            isCorrect: boolean | undefined
          }
        }
      | undefined
    subQuestionResults:
      | {
          [pollingSubQuestionID: string]: {
            [answerID: string]: {
              answerID: string
              answerValue: string | undefined
              frequency: number
              isCorrect: boolean | undefined
            }
          }
        }
      | undefined
  }
}
```

bar_chart

```
type PollQuestionItemData = {
  pollingID: string
  pollingQuestionID: string
  pollingQuestionName: string
  pollingQuestionType: 1 /* SingleSelect */ | 2 /* MultiSelect */
  answeredCount: number
  isRequired: boolean
  pollingSubQuestionItemList: {
    pollingParentQuestionID: string
    pollingSubQuestionID: string
    pollingQuestionName: string
  }[]
  questionImagePath: string | undefined
  visMethod: 'pie-chart' | 'bar-chart'
  graphicStatus: { showingResults: boolean; showingQuestion: boolean }
  result: {
    totalResponseCount: number | undefined
    questionResults:
      | {
          [answerID: string]: {
            answerID: string
            answerValue: string | undefined
            frequency: number
            isCorrect: boolean | undefined
          }
        }
      | undefined
    subQuestionResults:
      | {
          [pollingSubQuestionID: string]: {
            [answerID: string]: {
              answerID: string
              answerValue: string | undefined
              frequency: number
              isCorrect: boolean | undefined
            }
          }
        }
      | undefined
  }
}
```

pie_chart

```
type PollQuestionItemData = {
  pollingID: string
  pollingQuestionID: string
  pollingQuestionName: string
  pollingQuestionType: 1 /* SingleSelect */ | 2 /* MultiSelect */
  answeredCount: number
  isRequired: boolean
  pollingSubQuestionItemList: {
    pollingParentQuestionID: string
    pollingSubQuestionID: string
    pollingQuestionName: string
  }[]
  questionImagePath: string | undefined
  visMethod: 'pie-chart' | 'bar-chart'
  graphicStatus: { showingResults: boolean; showingQuestion: boolean }
  result: {
    totalResponseCount: number | undefined
    questionResults:
      | {
          [answerID: string]: {
            answerID: string
            answerValue: string | undefined
            frequency: number
            isCorrect: boolean | undefined
          }
        }
      | undefined
    subQuestionResults:
      | {
          [pollingSubQuestionID: string]: {
            [answerID: string]: {
              answerID: string
              answerValue: string | undefined
              frequency: number
              isCorrect: boolean | undefined
            }
          }
        }
      | undefined
  }
}
```

QA highlight graphic

question

```
type QuestionData = {
  questionID: string
  timestamp: number
  senderName: string | undefined
  questionContent: string
  isAnonymous: boolean
  upvoteNum: number
  liveAnsweringUserName: string | undefined
  isBeingLiveAnswered: boolean
  isMarkedAsAnswered: boolean
}
```

answer

```
type AnswerData = {
  questionID: string
  answerID: string
  timestamp: number
  answerContent: string
  senderName: string | undefined
  isLiveAnswer: boolean
}
```

QA scroll graphic

question

```
type QuestionData = {
  questionID: string
  timestamp: number
  senderName: string | undefined
  questionContent: string
  isAnonymous: boolean
  upvoteNum: number
  liveAnsweringUserName: string | undefined
  isBeingLiveAnswered: boolean
  isMarkedAsAnswered: boolean
}
```

answer

```
type AnswerData = {
  questionID: string
  answerID: string
  timestamp: number
  answerContent: string
  senderName: string | undefined
  isLiveAnswer: boolean
}
```

Active Speaker Name Tag

name_tag

```
<div class='name-tag-container-inner'>
  <div class='display-name-wrapper'>
    <span class='display-name'>{{Titlecase activeSpeakerName}}</span>
  </div>
</div>
<div class='backdrop'></div>
```

Chat scroll

chat_message example

```
<div class='template-chat-message-container'>
  {{#if senderAvatarPath}}
    <img class='sender-avatar' src='{{senderAvatarPath}}' />
  {{/if}}
  <div>
    <div>
      <span class='sender-display-name'>{{Titlecase (ShortenLastName senderDisplayName)}}</span>
      <time class='chat-timestamp'>{{FormattedTime timestamp 'p'}}</time>
    </div>
    <p class='msg-content'>{{msgContent}}</p>
  </div>
</div>
```

Chat highlight

chat_reply example

```
<div class='template-chat-comment-container'>
  <div class='parent-thread-reference-container'>
    <span class='replying-to-info'>
      Replying to thread by
      <span class='replying-to-display-name'>{{parentMessage.senderDisplayName}}</span>
    </span>
    <p class='parent-thread-msg-qoute'>{{parentMessage.msgContent}}</p>
  </div>
  <div class='comment-msg-container'>
    {{#if senderAvatarPath}}
      <img class='sender-avatar' src='{{senderAvatarPath}}' />
    {{/if}}
    <div>
      <div>
        <span class='sender-display-name'>{{Titlecase (ShortenLastName senderDisplayName)}}</span>
        <time class='chat-timestamp'>{{FormattedTime timestamp 'p'}}</time>
      </div>
      <p class='msg-content'>{{msgContent}}</p>
    </div>
  </div>
</div>
```

Closed Captions

caption_box

```
{{#each singleLanguageCaptions}}
  <div>
    <div>{{FormattedTime this.timeStamp 'p'}}</div>
    <div>Speaker name: {{this.speakerName}}</div>
    <div>Caption message content: {{this.messageContent}}</div>
  </div>
  <br />
{{/each}}
```

Emoji Reactions

fountain_box

```
<div class='emoji-page'>
  <div class='row emoji-box' id='emoji-box'>
    <div class='wrapper'>
      <div id='spawn-clap' class='clap emoji-icon'></div>
    </div>
    <div class='wrapper'>
      <div id='spawn-thumb' class='thumb emoji-icon'></div>
    </div>
    <div class='wrapper'>
      <div id='spawn-heart' class='heart emoji-icon'></div>
    </div>
    <div class='wrapper'>
      <div id='spawn-joy' class='joy emoji-icon'></div>
    </div>
    <div class='wrapper'>
      <div id='spawn-openmouth' class='openmouth emoji-icon'></div>
    </div>
    <div class='wrapper'>
      <div id='spawn-tada' class='tada emoji-icon'></div>
    </div>
  </div>
</div>
```

Poll result

question_title

```
<div class='poll-question-title-container'>
  <div class='title'>{{pollingQuestionName}}</div>
```

```
</div>
```

bar_chart

```
<div class="bar-chart-container">
  <div class="category-label-list-container">
    {{#each result.questionResults}}
    <div class="category-label">{{this.answerValue}}</div>
    {{/each}}
  </div>
  <div
    class="bar-list-container"
    style="--answer-item-count:{{Measurable-GetLength result.questionResults}}"
  >
    {{#each result.questionResults}}
    <div
      class="bar"
      style="--scale: calc(100%*({{this.frequency}})/max({{#each ../result.questionResults}}{{this.frequency}}, {{/each}}0));"
    ></div>
    {{/each}}
  </div>
</div>
```

pie_chart

```
<div class="pie-chart-container">
  <div class="pie-chat-result-container">
    {{#each result.questionResults}}
      <div class="pie-chart-slice" style="...">
        {{#if this.frequency}}
        <svg viewBox="0 0 200 200" style="...">
          <circle
            cx="100"
            cy="100"
            r="50"
            fill="none"
            stroke-width="100px"
            stroke-dasharray="314"
            stroke-dashoffset="calc(314 * (1 - {{this.frequency}}/{{../result.totalResponseCount}}))"
          />
        </svg>
        {{/if}}
      </div>
    {{/each}}
    {{#each result.questionResults}}
      <div class="percentage-label" style="...">
```

```
        <span style="...">{{Math-Multiply (Math-Divide this.frequency ../result.totalResponseCount 6) '100' 1}}%
      </div>
    {{/each}}
    </div>
  </div>
</div>
```

QA Highlight

answer example

```
<div class='template-answer-container'>
  {{#if senderAvatarPath}}
    <img class='sender-avatar' src='{{senderAvatarPath}}' />
  {{/if}}
  <div class='answer-text-container'>
    <p>Sender name: {{senderName}}</p>
    <p>Answer content: {{answerContent}}</p>
    <p>Timestamp: {{timestamp}}</p>
    <p>isLiveAnswer: {{isLiveAnswer}}</p>
  </div>
</div>
```

QA Scroll

question example

```
<div class='template-question-container'>
  <div>
    <div>
      {{#if senderName}}
        <span class='sender-display-name'>{{Titlecase senderName}}</span>
      {{else}}
        <span class='sender-display-name'>Anonymous</span>
      {{/if}}
      <time class='question-timestamp'>{{FormattedTime timestamp 'p'}}</time>
    </div>
    <p class='question-content'>{{questionContent}}</p>
  </div>
</div>
```

Handlebars

Overview

Meeting data is consumed using Handlebars. Handlebars provide built-in helpers that are used in the graphic templates. By leveraging these built-in Handlebars helpers, graphic templates can efficiently process and present meeting data in a variety of ways to meet different visualization needs.

Details of the Handlerbars templating language documentation can be found here

Custom helpers

The Zoom Graphic Toolkit provides defined helper functions to manipulate data. These helper functions include string manipulation, math operations, and other utility functions such as formatting timestamps.

ConvertNameToInitials

Convert the name into its first and last initials

```
function (text)
```

text - string
return - string

```
{{ConvertNameToInitials 'John Doe'}}

// Results: JD
```

Eq

Check if two arguments are equal

```
function (arg1, arg2)
```

arg1 - First argument of type any

arg2 - Second argument of type any

return - boolean

```
{{Eq 'option_1' 'option_2'}}

// Results: false
```

FormattedTime

Format the epoch timestamp to a long localized time in the format of HH:MM AM/PM using the pattern “p”
FormattedTime is a wrapper of the format() method from the date-fns library. The user can specify the format of the timestamp based on the formatStr pattern passed in. Refer to the date-fns library for a list of format patterns.

```
function (timestampStr, formatStr)
```

timestampStr - string (epoch time)

formatStr - string (a pattern from the date-fns library)

return - string

```
{{FormattedTime '1744399255000' 'p'}}
// Results: "12:20 PM"

{{FormattedTime '1744399255000' 'pp'}}
// Results: "12:20:55 PM"
```

Math-Add

Return the sum of two numbers while specifying the number of decimals to round

```
function (numAStr, numBStr, roundToDecimal)
```

numAStr - string

numBStr - string

roundToDecimal - number

return - string

```
{{Math-Add '4' '1' 0}}

// Results: "5"
```

Math-Compare

Compare if numAstr is greater than numBstr

```
function (numAStr, numBStr)
```

numAStr - string

numBStr - string

return - boolean

```
{{Math-Compare '4' '1'}}

// Results: true
```

Math-Cos

Return the cosine of a number while specifying the number of decimals to round

```
function (numStr, roundToDecimal)
```

numAStr - string (in radians)

roundToDecimal - number

return - string

```
{{Math-Cos '.4089' 4}}

// Results: "0.9176"
```

Math-Divide

Divide two numbers while specifying the number of decimals to round

```
function (numAStr, numBStr, roundToDecimal)
```

numAStr - string

numBStr - string

roundToDecimal - number

return - string

```
{{Math-Divide '7' '3' 1}}

// Results: "2.3"
```

Math-Multiply

Multiply two numbers while specifying the number of decimals to round

```
function (numAStr, numBStr, roundToDecimal)
```

numAStr - string

numBStr - string

roundToDecimal - number

return - string

```
{{Math-Multiply '10' '.75' 1}}
// Results: 7.5
```

Math-Sin

Return the sine of a number while specifying the number of decimals to round

```
function (numStr, roundToDecimal)
```

numAStr - string (in radians)

roundToDecimal - number

return - string

```
{{Math-Sin '.4089' 4}}

// Results: "0.3976"
```

Math-Subtract

Return the difference of two numbers while specifying the number of decimals to round

```
function (numAStr, numBStr, roundToDecimal)
```

numAStr - string
numBStr - string
roundToDecimal - number
return - string

```
{{Math-Subtract '4' '1' 0}}

// Results: "3"
```

Measurable-GetLength

Get the length of a string or the number of keys in an object or the length of an array

```
function (obj)
```

obj - object or array or string
return - string

SelectPseudoRandomly

Randomly select an item from the passed in arguments

```
function (...args)
```

args - All of the arguments passed in using the handler bar helper (see example). The first argument is a string that is used to generate a random seed. The random seed is then used to randomly select the rest of the arguments
return - string

```
{{SelectPseudoRandomly 'TestSeed' 'foo' 'bar' 'baz'}}

// Result: "bar"
```

ShortenLastName

Shorten the string of names to only the first full name and last name initial

```
function (text)
```

text - string
return - string

String-Split

Split a string into substrings using the specified separator and return them as an array

```
function (str, separator)
```

str - string
separator - string. It identifies character or characters to use in separating the string
return - array of strings

```
{{String-Split 'Hello World' ' '}}

// Results: ["Hello", "World"]
```

Titlecase

Uppercase the beginning of every word of the string

```
function (text)
```

text - string

return - string

Framer Motion Presence Animation CSS Hooks

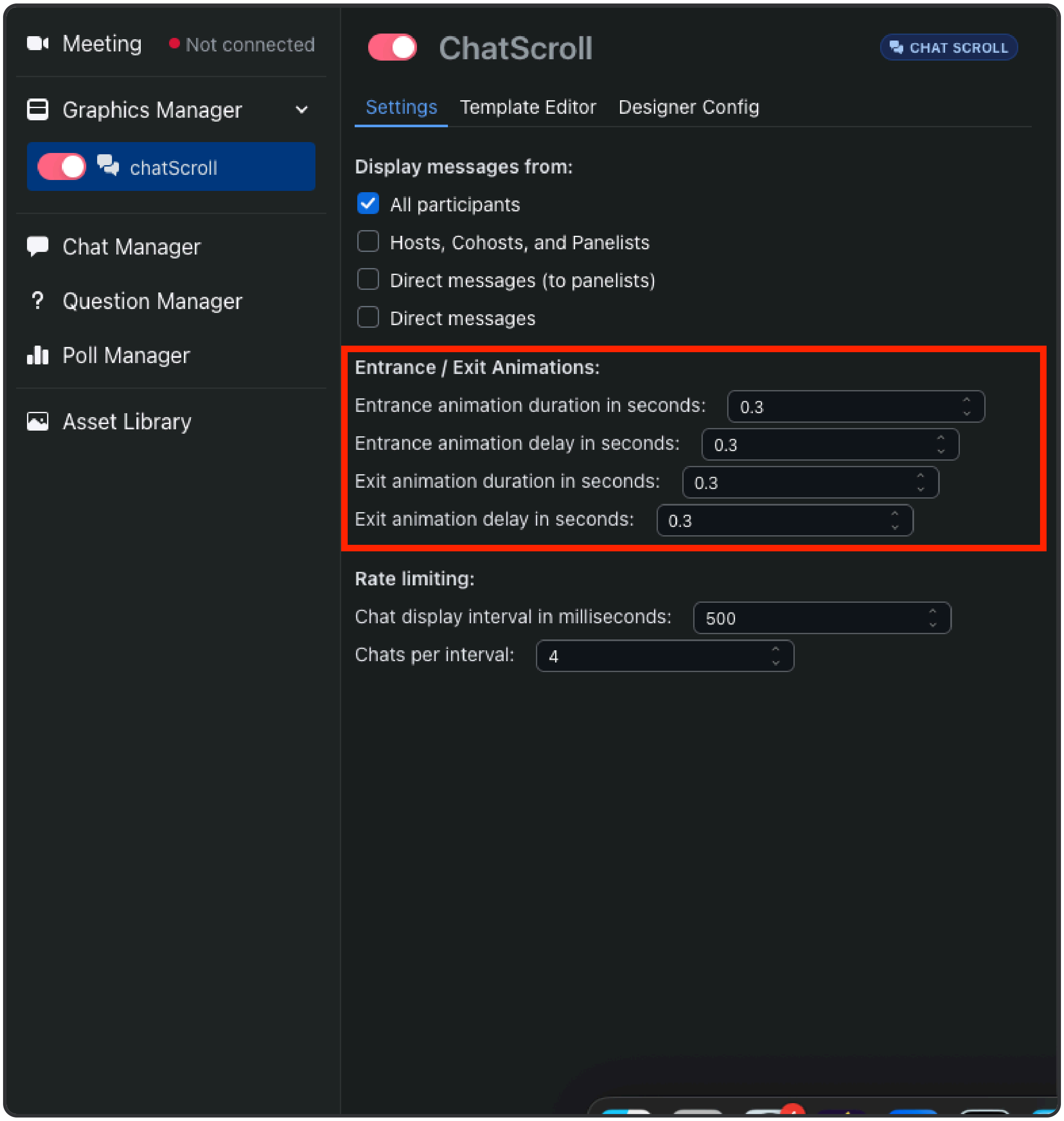
The Zoom Graphics Toolkit uses Framer Motion's **AnimatePresence** to animate components in and out of view across different graphic types.

This means that every time a graphic is:

- **Rendered** → an **entrance animation** runs.
- **Removed** → an **exit animation** runs.

Timing Configuration and Constraints

- You can customize the entrance/exit timing in the graphic designer's settings panel:



Display messages from:

- ☑ All participants
- ☐ Hosts, Cohosts, and Panelists
- ☐ Direct messages (to panelists)
- ☐ Direct messages

Entrance / Exit Animations:

Entrance animation duration in seconds: 0.3

Entrance animation delay in seconds: 0.3

Exit animation duration in seconds: 0.3

Exit animation delay in seconds: 0.3

Rate limiting:

Chat display interval in milliseconds: 500

Chats per interval: 4

- Duration constraints: 0.0 seconds to 10.0 seconds
- Delay constraints: 0.0 seconds to 10.0 seconds

Parent Containers

All HTML written by the user in the *Template Editor* tab exists within a parent div of:

- **chat-message-container** or **chat-reply-container** (for Chat Scroll and Chat Highlight graphics)

- **answer-message-container** or **question-message-container** (for QA Scroll and QA Highlight graphics)

Those divs also exist within a parent div of **content-area** (regardless of graphic type). It is important to implement styling in the CSS for the automatically injected divs in order for animations to apply.

The following is an example of a chat highlight graphic that manipulates these parent containers.

```
<div class='chat-highlight-lower-third'>
  {{#if senderAvatarPath}}
    <img class='chat-avatar' src='{{senderAvatarPath}}' alt='Avatar' />
  {{/if}}
  <div class='chat-content'>
    <div class='chat-header'>
      <span class='sender-name'>{{Titlecase senderDisplayName}}</span>
      <time class='chat-time'>{{FormattedTime timestamp 'p'}}</time>
    </div>
    <p class='chat-message'>{{msgContent}}</p>
  </div>
</div>
```

```
.content-area {
  height: 100%;
}

.chat-message-container {
  position: relative;
  height: 100%;
  width: 100%;
}

.chat-reply-container {
  position: relative;
  height: 100%;
  width: 100%;
}

.chat-highlight-lower-third {
  position: absolute;
  bottom: 10%;
  left: 50%;
  transform: translateX(-50%);
  background-color: var(--background-color, #000000aa);
  color: var(--text-color, #ffffff);
```

```css
  padding: var(--padding, 10px);
  border-radius: var(--border-radius, 5px);
  font-family: var(--font-family, 'Almaden Sans'), sans-serif;
  font-size: var(--font-size, 14px);
  display: flex;
  align-items: center;
  max-width: 80%;
  box-shadow: 0 2px 10px rgba(0, 0, 0, 0.3);
}
```

Note that in the above example, it is critical that the CSS addresses style for **`.content-area`**, **`.chat-message-container`** and **`.chat-reply-container`** in order for the Motion animations to apply.

CSS Hooks

Users can add additional animation styling using the *framer-animate-presence-exiting* class which is applied to an element that is about to be removed.

The following is an example of how a poll graphic type might use this class.

```css
.question-title-container.framer-animate-presence-exiting {
  animation-name: slide-out;
  animation-delay: 0s;
  animation-duration: 0.5s;
  animation-timing-function: ease-in;
  animation-fill-mode: both;
  animation-direction: normal;
}
```

Here’s another example, this time with a chat scroll graphic type with a custom animation.

```css
.template-chat-message-container.framer-animate-presence-exiting {
  animation-name: fade-slide-out;
  animation-duration: 0.5s;
  animation-fill-mode: both;
  animation-timing-function: ease-in;
}

@keyframes fade-slide-out {
  0% {
    transform: translateX(0);
    opacity: 1;
  }
  100% {
    transform: translateX(100%);
    opacity: 0;
  }
}
```

Syntax

Graphic designer form generation

Syntax

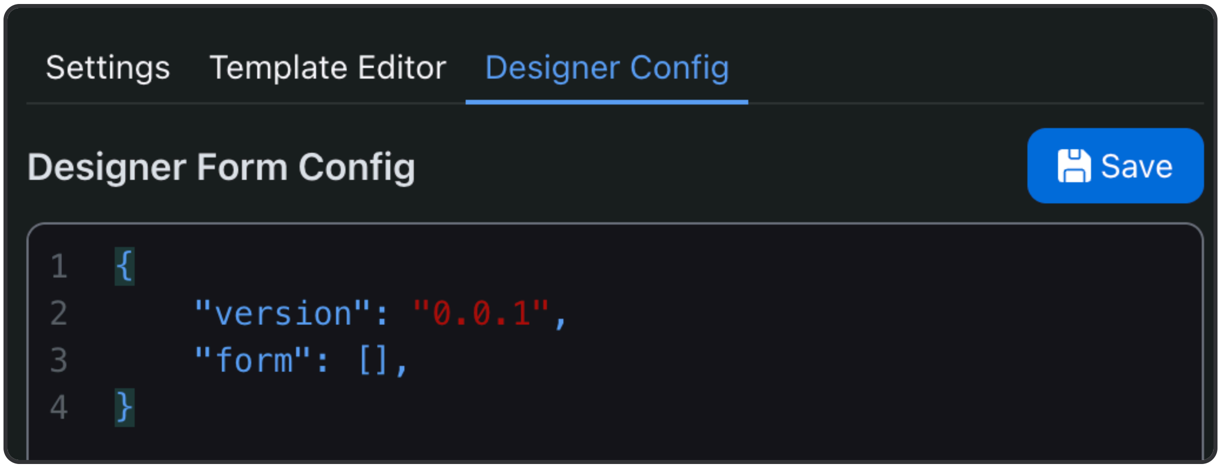
Since the graphic designer form configuration is represented as a JSON format, users must adhere to JSON syntax rules.

- keys and values must be enclosed with double quotes NOT single quotes
- The version number must be a semantic version number. An example would be “1.0.0”. (see https://semver.org/ for more information).

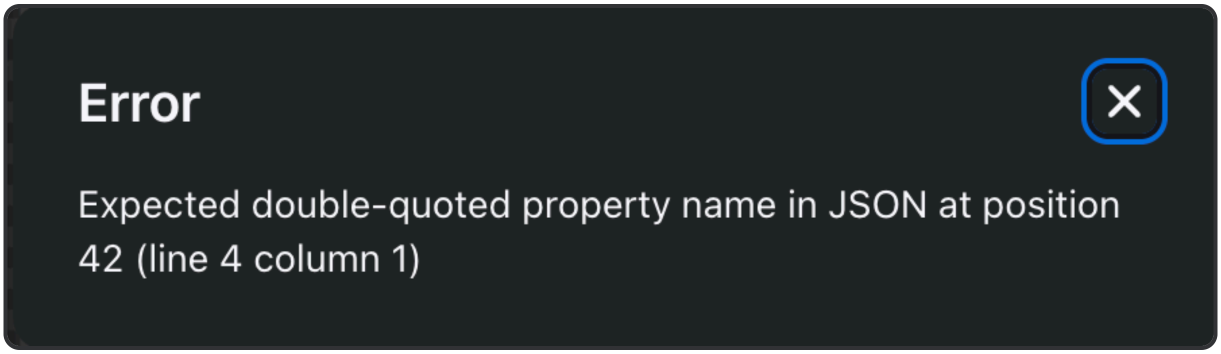
```
// This is correct ✅
"version": "1.0.0"
"form-checkbox": "true"
"max": "40"

// This is wrong 🚫
"version": 1.0.0
"version": "v1.0"
version: "1.0.0"
'version': "1.0.0"
'version': '1.0.0'
"form-checkbox": true
"max": 40
```

- JSON data must be followed by a comma when there is more data after it, otherwise you will encounter an error that prevents saving your changes

```
// This is correct ✅
{
   "version": "0.0.1",
   "form": [...]
}

// This is correct ✅
{
   "version": "0.0.1",
   "form": [...],
   "sectionMacros": {...}
}

// This is wrong 🚫
{
   "version": "0.0.1",
   "form": [...],
// There is no data after "form" so this comma will cause an error
}

```



```
{
    "version": "0.0.1",
    "form": [],
}
```



Error

Expected double-quoted property name in JSON at position 42 (line 4 column 1)

- Escape the quotation character "" using the backslash \ if you want to include it part of your string

```
// This is correct ✅
```

```
"css-image": "url(\"\")"

// This is wrong 🚫
"css-image": "url("")"
```

Configuration property references

- The code snippets and tables below serve as references for the properties that can be utilized in the designer form configuration
- Form vs Macro
  - **Form**: The general layout of what the design form will include
  - **Macro**: The pre-programmed pieces of the design form that allows you to adjust the designs of your graphic. With these macros, you can easily update colors, sizes, layouts, and other design properties without rebuilding the entire graphic from scratch.
    - The benefit of setting parameterMacros and sectionMacros is that they are re-usable. The macros can be used multiple times without having to re-create a GraphicDesignerFormSection object or GraphicDesignerParameter inside the form

Design form configuration

Typescript definition

```
type GraphicDesignerFormConfig = {
  version: string
  parameterMacros?: { [macroName: string]: GraphicDesignerParameter } // optional
  sectionMacros?: { [macroName: string]: GraphicDesignerFormSection } // optional
  form: (GraphicDesignerFormSection | SectionMacroReference)[]
}
```

JSON example

```
{
  "version": "1.0.0",
  "form": ["BACKGROUND"]
}
```

Table reference

Property	Type	Required
version	string	
parameterMacros	`{ [ macroName: string ]: GraphicDesignerParameter }`	optional
sectionMacros	`{ [ macroName: string ]: GraphicDesignerFormSection }`	optional
form	Array of GraphicDesignerFormSection and SectionMacroReference	

Default macro sections

The following list are names (macroName) for default macro presets that the user can choose from. Each of the presets follows the syntax of the GraphicDesignerFormSection object.

- BOUNDS

```
{
  "label": "Bounds",
  "slug": "bounds",
  "subSectionsPresentation": "column",
  "subsections": [
    {
      "slug": "vertical",
      "parametersPresentation": "row",
      "parameters": [
        {
          "label": "Vertical Anchor",
          "slug": "anchor",
          "types": ["gfx-global-variable", "form-select"],
          "defaults": {
            "form-select": "top"
          },
          "defaultType": "form-select",
          "parameterTypeBasedOptions": {
            "form-select": {
              "options": [
                {
                  "label": "Top & Bottom",
                  "value": "top-and-bottom"
                },
                {
                  "label": "Top",
                  "value": "top"
                },
                {
                  "label": "Bottom",
                  "value": "bottom"
                },
                {
                  "label": "Center",
                  "value": "center"
                }
              ]
            }
          }
        },
        {
          "label": "Top",
          "slug": "top",
          "types": ["gfx-global-variable", "css-length", "css-percentage"],
          "defaults": {
            "css-length": "0px",
            "css-percentage": "0%"
          },
          "defaultType": "css-length",
          "dependencies": [
            {
              "accessor": "./anchor",
              "values": ["top", "top-and-bottom"]
            }
          ]
        },
        {
          "label": "Translate-y",
          "slug": "translate_y",
          "types": ["gfx-global-variable", "css-length", "css-percentage"],
          "defaults": {
            "css-length": "0px",
            "css-percentage": "0%"
          },
          "defaultType": "css-length",
          "dependencies": [
            {
              "accessor": "./anchor",
              "values": ["center"]
            }
          ]
        },
        {
          "label": "Bottom",
          "slug": "bottom",
          "types": ["gfx-global-variable", "css-length", "css-percentage"],
          "defaults": {
            "css-length": "0px",
            "css-percentage": "0%"
          },
          "defaultType": "css-length",
          "dependencies": [
            {
              "accessor": "./anchor",
              "values": ["bottom", "top-and-bottom"]
            }
          ]
        },
        {
          "label": "Height",
          "slug": "height",
          "types": ["gfx-global-variable", "css-length", "css-percentage", "form-select"],
          "defaults": {
            "css-length": "0px",
            "css-percentage": "0%"
          },
          "defaultType": "css-length",
          "dependencies": [
            {
              "accessor": "./anchor",
              "values": ["top", "bottom", "center"]
            }
          ],
          "parameterTypeBasedOptions": {
            "form-select": {
              "options": ["auto", "max-content", "min-content", "fit-content", "stretch"]
            }
          }
        }
      ]
    },
    {
      "slug": "horizontal",
      "parametersPresentation": "row",
      "parameters": [
        {
          "label": "Horizontal Anchor",
          "slug": "anchor",
          "types": ["gfx-global-variable", "form-select"],
          "defaults": {
            "form-select": "left"
          },
          "defaultType": "form-select",
          "parameterTypeBasedOptions": {
            "form-select": {
              "options": [
                {
                  "label": "Left & Right",
                  "value": "left-and-right"
                },
                {
                  "label": "Left",
                  "value": "left"
                },
                {
                  "label": "Right",
                  "value": "right"
                },
                {
                  "label": "Center",
                  "value": "center"
                }
              ]
            }
          }
        },
        {
          "label": "Left",
          "slug": "left",
          "types": ["gfx-global-variable", "css-length", "css-percentage"],
          "defaults": {
            "css-length": "0px",
            "css-percentage": "0%"
          },
          "defaultType": "css-length",
          "dependencies": [
            {
              "accessor": "./anchor",
              "values": ["left", "left-and-right"]
            }
          ]
        },
        {
          "label": "Right",
          "slug": "right",
          "types": ["gfx-global-variable", "css-length", "css-percentage"],
          "defaults": {
            "css-length": "0px",
            "css-percentage": "0%"
          },
          "defaultType": "css-length",
          "dependencies": [
            {
              "accessor": "./anchor",
              "values": ["right", "left-and-right"]
            }
          ]
        },
        {
          "label": "Translate-x",
          "slug": "translate_x",
          "types": ["gfx-global-variable", "css-length", "css-percentage"],
          "defaults": {
            "css-length": "0px",
            "css-percentage": "0%"
          },
          "defaultType": "css-length",
          "dependencies": [
            {
              "accessor": "./anchor",
              "values": ["center"]
            }
          ]
        },
        {
          "label": "Width",
          "slug": "width",
          "types": ["gfx-global-variable", "css-length", "css-percentage", "form-select"],
          "defaults": {
            "css-length": "0px",
            "css-percentage": "0%"
          },
          "defaultType": "css-length",
          "dependencies": [
            {
              "accessor": "./anchor",
              "values": ["left", "right", "center"]
            }
          ],
          "parameterTypeBasedOptions": {
            "form-select": {
              "options": ["auto", "max-content", "min-content", "fit-content", "stretch"]
            }
          }
        }
      ]
    }
  ]
}
```

- SIZE_CONSTRAINTS

```
{
  "label": "Size Constraints",
  "slug": "size_constraint",
  "parametersPresentation": "row",
  "parameters": [
    {
      "label": "Min-Width",
      "slug": "min_width",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "0%"
      },
      "defaultType": "css-length"
    },
    {
      "label": "Max-Width",
      "slug": "max_width",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "100%"
      },
      "defaultType": "css-percentage"
    },
    {
      "label": "Min-Height",
      "slug": "min_height",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "0%"
      },
      "defaultType": "css-length"
    },
    {
      "label": "Max-Height",
      "slug": "max_height",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "100%"
      },
      "defaultType": "css-percentage"
    }
  ]
}
```

- PADDING

```json
{
  "label": "Paddding",
  "slug": "padding",
  "parametersPresentation": "row",
  "parameters": [
    {
      "label": "Top",
      "slug": "top",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "0%"
      },
      "defaultType": "css-length"
    },
    {
      "label": "Right",
      "slug": "right",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "0%"
      },
      "defaultType": "css-length"
    },
    {
      "label": "Bottom",
      "slug": "bottom",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "0%"
      },
      "defaultType": "css-length"
    },
    {
      "label": "Left",
      "slug": "left",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "0%"
      },
      "defaultType": "css-length"
    }
  ]
}
```

- MARGIN

```json
{
  "label": "Margin",
  "slug": "margin",
  "parametersPresentation": "row",
  "parameters": [
    {
      "label": "Top",
      "slug": "top",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "0%"
      },
      "defaultType": "css-length"
    },
    {
      "label": "Right",
      "slug": "right",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "0%"
      },
      "defaultType": "css-length"
    },
    {
      "label": "Bottom",
      "slug": "bottom",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "0%"
      },
      "defaultType": "css-length"
    },
    {
      "label": "Left",
      "slug": "left",
      "types": ["gfx-global-variable", "css-length", "css-percentage"],
      "defaults": {
        "css-length": "0px",
        "css-percentage": "0%"
      },
      "defaultType": "css-length"
    }
  ]
}
```

- TEXT_STYLING

```json
{
  "label": "Text Styling",
  "slug": "text_style",
  "subSectionsPresentation": "column",
  "subsections": [
    {
      "slug": "font",
      "parametersPresentation": "row",
      "parameters": [
        {
          "slug": "family",
          "label": "Font Family",
          "types": ["gfx-global-variable", "css-font-family"],
          "defaults": {
            "css-font-family": "system-ui"
          }
        },
        {
          "slug": "size",
          "label": "Font Size",
          "types": ["gfx-global-variable", "css-length", "css-percentage"],
          "defaults": {
            "css-length": "14px",
            "css-percentage": "0%"
          },
          "defaultType": "css-length"
        },
        {
          "slug": "weight",
          "label": "Font Weight",
          "types": ["gfx-global-variable", "form-select", "form-number"],
          "defaults": {
            "form-select": "400",
            "form-number": "400"
          },
          "defaultType": "form-select",
          "parameterTypeBasedOptions": {
            "form-select": {
              "options": [
                {
                  "label": "Thin",
                  "value": "100"
                },
                {
                  "label": "Extra Light",
                  "value": "200"
                },
                {
                  "label": "Light",
                  "value": "300"
                },
                {
                  "label": "Normal",
                  "value": "400"
                },
                {
                  "label": "Medium",
                  "value": "500"
                },
                {
                  "label": "Semi Bold",
                  "value": "600"
                },
                {
                  "label": "Bold",
                  "value": "700"
                },
                {
                  "label": "Extra Bold",
                  "value": "800"
                },
                {
                  "label": "Black",
                  "value": "900"
                }
              ]
            }
          }
        }
      ]
    },
    {
      "slug": "format",
      "parametersPresentation": "row",
      "parameters": [
        {
          "slug": "is_italicized",
          "label": "Italicize",
          "types": ["gfx-global-variable", "form-checkbox"],
          "defaults": {
            "form-checkbox": "true"
          }
        },
        {
          "slug": "is_underlined",
          "label": "Underline",
          "types": ["gfx-global-variable", "form-checkbox"],
          "defaults": {
            "form-checkbox": "true"
          }
        },
        {
          "slug": "color",
          "label": "color",
          "types": ["gfx-global-variable", "css-color"],
          "defaults": {
            "form-checkbox": "#000000FF"
          }
        }
      ]
    }
  ]
}
```

- BORDER_STYLING

```
{
  "slug": "border",
  "label": "Border Styling",
  "parametersPresentation": "row",
  "parameters": [
    {
      "label": "Border Thickness",
      "slug": "thickness",
      "types": ["gfx-global-variable", "css-length"],
      "defaults": {
        "css-length": "0px"
      }
    },
    {
      "label": "Border Radius",
      "slug": "radius",
      "types": ["gfx-global-variable", "css-length"],
      "defaults": {
        "css-length": "0px"
      }
    },
    {
      "label": "Border Color",
      "slug": "color",
      "types": ["gfx-global-variable", "css-color"],
      "defaults": {
        "css-color": "#000000FF"
      }
    }
  ]
}
```

- SHADOW_STYLING

```
{
  "slug": "shadow",
  "label": "Shadow Styling",
  "subSectionsPresentation": "column",
  "subsections": [
    {
      "slug": "",
      "parametersPresentation": "row",
      "parameters": [
        {
          "slug": "is_visible",
          "label": "Show Shadow",
          "types": ["gfx-global-variable", "form-checkbox"],
          "defaults": {
            "form-checkbox": "false"
          }
        },
        {
          "slug": "color",
          "label": "Shadow Color",
          "types": ["gfx-global-variable", "css-color"],
          "defaults": {
            "css-color": "#000000FF"
          }
        }
      ]
    },
    {
      "slug": "",
      "parametersPresentation": "row",
      "parameters": [
        {
          "slug": "x_offset",
          "label": "X Offset",
          "types": ["gfx-global-variable", "css-length"],
          "defaults": {
            "css-length": "0px"
          }
        },
        {
          "slug": "y_offset",
          "label": "Y Offset",
          "types": ["gfx-global-variable", "css-length"],
          "defaults": {
            "css-length": "0px"
          }
        },
        {
          "slug": "blur_radius",
          "label": "Blur Radius",
          "types": ["gfx-global-variable", "css-length"],
          "defaults": {
            "css-length": "0px"
          }
        }
      ]
    }
  ]
}
```

- BACKGROUND

```
{
  "label": "Background",
  "slug": "background",
  "parameters": [
    {
      "label": "Fill",
      "slug": "fill",
      "types": ["gfx-global-variable", "css-color", "css-image"],
      "defaults": {
        "css-color": "#000000FF",
        "css-image": "url(\"\")"
      },
      "defaultType": "css-color"
    }
  ],
  "parametersPresentation": "row",
  "subSectionsPresentation": "column",
  "subsections": [
    {
      "slug": "",
      "parametersPresentation": "row",
      "parameters": [
        {
          "label": "Background Size",
          "slug": "size",
          "types": ["gfx-global-variable", "form-select", "css-length", "css-percentage"],
          "parameterTypeBasedOptions": {
            "form-select": {
              "options": ["cover", "contain", "auto"]
            }
          },
          "defaults": {
            "form-select": "contain"
          },
          "defaultType": "form-select"
        },
        {
          "label": "Background Repeat",
          "slug": "repeat",
          "types": ["gfx-global-variable", "form-select"],
          "parameterTypeBasedOptions": {
            "form-select": {
              "options": ["repeat", "no-repeat", "space", "round", "repeat-x", "repeat-y"]
            }
          },
          "defaults": {
            "form-select": "repeat"
          },
          "defaultType": "form-select"
        }
      ]
    }
  ]
}
```

If a user wanted to, they can create a custom macro with the macroName of CUSTOM10 and they can copy the value of BOUNDS and paste for the value of CUSTOM10 inside the sectionMacros property.

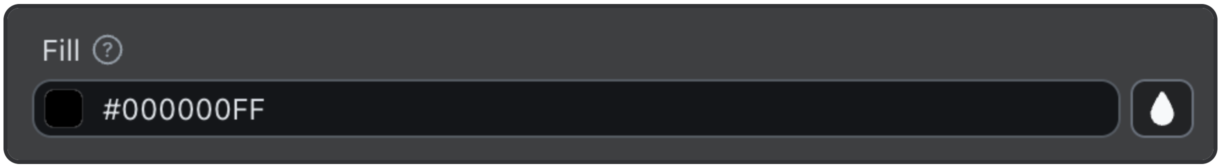
GraphicDesignerParameter object

Typescript definition

```
type GraphicDesignerParameter = {
  label?: string // optional
  slug: string
  description?: string // optional
  types: GraphicDesignerParameterType[]
  defaultType?: GraphicDesignerParameterType // optional
  defaults?: {
    [type in GraphicDesignerParameterType]?: string // optional
  } // optional
  parameterTypeBasedOptions?: {
    'css-length'?: CssLengthParamOptions // optional
    'css-percentage'?: CssPercentageOptions // optional
    'css-angle'?: CssAngleOptions // optional
    'css-color'?: CssColorOptions // optional
    'css-image'?: CssImageOptions // optional
    'form-text'?: FormTextOptions // optional
    'form-number'?: FormNumberOptions // optional
    'form-select'?: FormSelectOptions // optional
    'form-checkbox'?: FormCheckboxOptions // optional
    'form-toggle'?: FormToggleOptions // optional
  } // optional
  dependencies?: {
    accessor: string
    values: string[] // or
  }[] // optional // and
}
```

JSON example

```
{
  "label": "Fill",
  "slug": "fill",
  "types": ["css-color", "css-image"],
  "defaults": {
    "css-color": "#000000FF",
    "css-image": "url(\"\")"
  },
  "defaultType": "css-color"
}
```



Fill ?

#000000FF

Table reference

Property	Type	Required
label	string	optional
slug	string	
description	string	optional
types	Array of GraphicDesignerParameterType	
defaultType	GraphicDesignerParameterType	optional
defaults	*Refer to code snippet*	optional
parameterTypeBasedOptions	*Refer to code snippet*	optional
dependencies	*Refer to code snippet*	optional

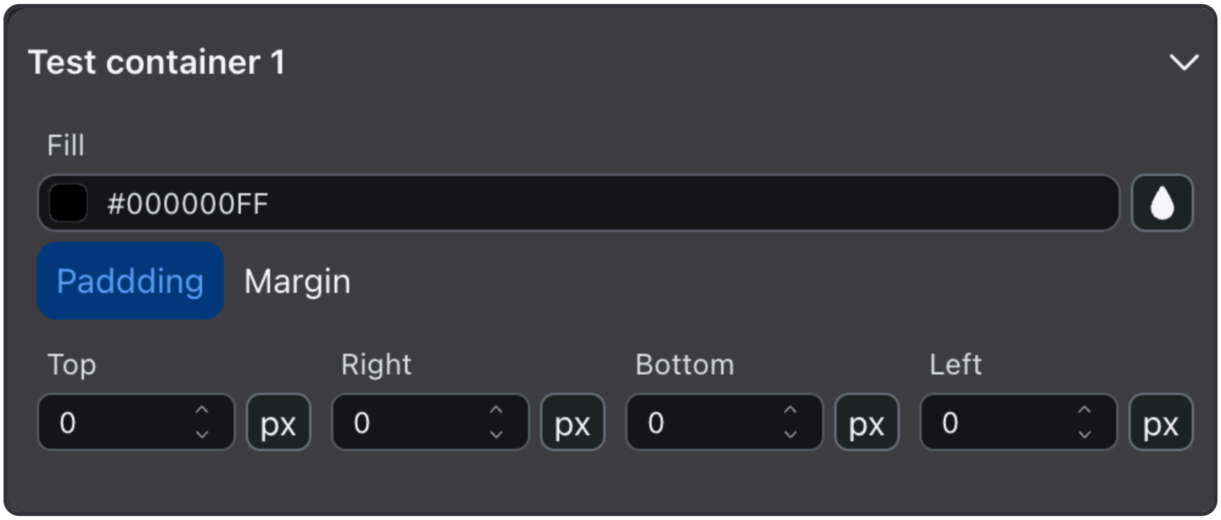
GraphicDesignerFormSection object

Typescript definition

```typescript
type GraphicDesignerFormSection = {
  label?: string // optional
  slug: string
  description?: string // optional
  parametersPresentation?: 'flex-grid' | 'column' | 'row' // optional
  parameters?: (GraphicDesignerParameter | ParameterMacroReference)[] // optional
  subSectionsPresentation?: 'accordion' | 'tabs' | 'column' | 'row' // optional
  subsections?: (GraphicDesignerFormSection | SectionMacroReference)[] // optional
}
```

JSON example

```json
{
  "label": "Test container 1",
  "slug": "tc_1",
  "description": "This is a test GraphicDesignerFormSection",
  "parametersPresentation": "column",
  "parameters": [
    {
      "label": "Fill",
      "slug": "fill",
      "types": ["css-color", "css-image"],
      "defaults": {
        "css-color": "#000000FF",
        "css-image": "url(\"\")"
      },
      "defaultType": "css-color"
    }
  ],
  "subSectionsPresentation": "tabs",
  "subsections": ["PADDING", "MARGIN"]
}
```



Test container 1

Fill

#000000FF

Paddding Margin

Top 0 px

Right 0 px

Bottom 0 px

Left 0 px

Table Reference

Property	Type	Required
label	string	optional
slug	string	
description	string	optional
parametersPresentation	"flex-grid"	"column" | "row" | optional
parameters	Array of GraphicDesignerParameter and ParameterMacroReference	optional
subSectionsPresentation	"accordion"	"tabs" | "column" | "row" | optional
subsections	Array of GraphicDesignerFormSection and SectionMacroReference	optional

SectionMacroReference type

This is a reference to either a default **section** macro preset or a custom section macro created by the user. The user can either provide the macro name as a simple string or supply an object to overwrite both the label and slug.

Typescript definition

```
type SectionMacroReference =
  | string
  | {
      macroName: string
      label?: string
      slug?: string
    }
```

Table reference

Property	Type	Required
macroName	string	"BOUNDS" | "SIZE_CONSTRAINT" | "PADDING" | "MARGIN" | "TEXT_STYLING" | "BORDER_STYLING" | "SHADOW_STYLING" | "BACKGROUND"
label	string	optional
slug	string	optional

ParameterMacroReference object

This is a reference to a custom **parameter** macro created by the user. The user can either provide the macro name as a simple string or supply an object to overwrite both the label and slug.

Typescript definition

```
type ParameterMacroReference =
  | string
  | {
      macroName: string
      label?: string
      slug: string // optional
    }
```

Table reference

Property	Type	Required
macroName	string	
label	string	optional

Property	Type	Required
slug	string	optional

Reference

GraphicDesignerParameterType options

Table reference

Option name	Type
css-length	string
css-percentage	string
css-angle	string
css-color	string
css-image	string
css-font-family	string
form-text	string
form-number	string
form-select	string
form-checkbox	string
form-toggle	string
gfx-global-variable	string

CSS Parameter options

CssLengthParamOptions

Typescript definition

```
type CssLengthParamOptions = {
  units: ('px' | 'em' | 'vh' | 'vw' | 'cqw' | 'cqh' | 'cqi')[] // empty means all
  constraints?: {
    [unit in 'px' | 'em' | 'vh' | 'vw' | 'cqw' | 'cqh' | 'cqi']: {
      min?: number // optional
      max?: number // optional
    }
  }[] // optional
}
```

Table reference

Property	Type	Required
units	Array of "px" | "em" | "vh" | "vw" | "cqw" | "cqh" | "cqi"	
constraints	*Refer to code snippet*	optional

CssPercentageOptions

Typescript definition

```
type CssPercentageOptions = {
  constraints?: {
    min?: number
    max?: number
  }
}
```

Table reference

Property	Type	Required
constraints	*Refer to code snippet*	optional

CssAngleOptions

No support yet

CssColorOptions

Typescript definition

```
undefined
```

CssImageOptions

Typescript definition

```
undefined
```

CssFontFamily

Typescript definition

```
undefined
```

Form parameter options

FormTextOptions

Typescript definition

```
undefined
```

FormNumberOptions

Typescript definition

```
undefined
```

FormSelectOptions

Typescript definition

```
type FormSelectOptions = {
  options:
    | {
        value: string
        label: string
      }[]
    | string[]
}
```

JSON example

```json
{
  "options": [
    {
      "value": "500",
      "label": "Medium"
    }
  ]
}
```

Table reference

Property	Type	Required
options	Array of string or Array of `` `{"value": string, "label": string}` ``	

FormCheckboxOptions

Typescript definition

```
undefined
```

FormToggleOptions

Typescript definition

```
undefined
```

Special Gfx parameter options

GfxGlobalVariableOptions

Typescript definition

```
undefined
```

Combiner Graphic Type

Overview

The Combiner graphic type is a unique graphic type in the Zoom Graphics Toolkit. It allows the user to combine multiple existing graphic layers into one output, controlling placement and scale.

Creating A Combiner Graphic

You can select the combiner graphic type the same way as all other graphic types, by clicking New Graphic in the Graphics Dashboard. You can then select the "Combiner" graphic type from the displayed list.

The Combiner graphic type currently only supports one, default "Blank" preset, and creating new presets is not currently supported.

Add Graphics To Your Combiner

Upon creating a combiner graphic, you'll be greeted with a Graphics Editor page similar to that of other graphic types.

You may notice that within the left hand side of the Graphics Editor, you have only one tab labeled "Combiner". From here, you can add some of your existing graphics to be displayed on this Combiner graphic by clicking the "Add Graphic" button.

Upon clicking the "Add Graphic" button, you'll see a dialog box containing a list of all your already created graphics, excluding any of your combiner graphics.

You can check the checkbox besides any of those listed graphics to add them to your Combiner graphic. When finished with your selection, you can press the "Update Selection" button to proceed.

Please note that you can only have a maximum of 5 "component" graphics displayed on a single Combiner graphic at a time.

An Overview of the "Component" Graphic Settings Boxes

After clicking the "Update Selection" button in the Add Graphic dialog, you'll see a collapsed settings box appear for each of your selected graphics. We'll refer to these selected graphics as "component graphics" of the larger "Combiner graphic".

Each box will display the graphic type icon and the graphic name of the component graphic on the left hand side, along with some settings buttons on the right hand side.

Some details on those settings buttons:

- The eye icon will toggle the visiblity of that component graphic on the combiner graphic. **Please note this only affects how it displays on this exact combiner graphic, and does not toggle its visibility elsewhere in the app.**
- The arrow icon will expand and collapse the settings box to show or hide further controls.
- The ... icon will allow you to access other controls for the graphic, such as removing the graphic from the Combiner.

Expanding The Settings Box and Starting A Simulation For A Component Graphic

Try clicking on the expand / collapse arrow on the right hand side of one of your component graphic's settings box.

Here, you'll see a few options for manipulating the width, height, x position, y position, and scale for your graphic. **Please note these controls only affect these properties of the component graphic within this specific Combiner graphic, and does not affect it anywhere else in the app.**

For now, take note of the "Open Editor" and "Simulation Settings" buttons.

Clicking the "Open Editor" button will redirect you to the full Graphics Editor for that specific graphic.

Clicking the "Simulation Settings" will open a pop up dialog containing just the simulation tools for that specific graphic. **Please note that the "Show simulated data on live output" checkbox is set and locked to on in this window, since sim data would otherwise not be visible from the Combiner graphic.**

Try starting a simulation for a component graphic you've selected. If that graphic was empty before, it will now populate with simulated data and be visible.

Manipulating The Position and Scale Of A Component Graphic

Now try changing the width, height, x or y positions, or scale of one of your component graphics. You can do this by pressing the up / down arrows for each property or by typing in your own values.

Press enter or click off the property's input box to see your changes take effect.

Reordering the Z-Index of Your Component Graphics

Finally, you can reorder the z-index of your component graphics on the Combiner by simply dragging and dropping each graphic's settings box to the position you'd like it to be.

The top graphic will be displayed on the top (in other words, become the highest z-index value).

The bottom graphic will be displayed on the bottom (becoming the lowest z-index value).

The page may pause and refresh while reordering graphics, and reappear shortly after with updated z-index ordering.