



JL Web Framework

Руководство программиста
ПЛАБ.421001.004-33



г. Пенза

2019



Содержание

Содержание.....	2
1 Введение	3
2 Использование библиотеки	4
2.1 Установка	4
2.2 Минимальный шаблон страницы HTML.....	4
2.3 Связь с переменными во внешних устройствах.....	5
2.4 Глобальные переменные.....	5
2.5 Создание объектов из кода JS	7
2.6 Атрибуты отображения объектов.....	7
2.7 Событийная модель воздействий.....	7
2.8 Динамическое изменение атрибутов компонента.....	8
3 Графические компоненты	9
3.1 Окружность	9
3.2 Эллипс.....	11
3.3 Прямоугольник.....	13
3.4 Линия.....	15
3.5 Ломанная	17
3.6 Полигон	19
3.7 Текст.....	21
3.8 Изображение.....	23
3.9 Графический индикатор и переключатель битовых состояний	25
3.10 Слайдер	27
3.11 Вертикальный слайдер	29
3.12 Полоса состояния.....	32
3.13 Вертикальная полоса состояния.....	34
3.14 Битовый переключатель.....	37
3.15 Вертикальный битовый переключатель	39
3.16 Кнопка	42
3.17 Графическая кнопка	44
3.18 Круговая диаграмма.....	47
3.19 Кольцевая диаграмма.....	50
3.20 Столбчатая диаграмма	53
3.21 Блок вывода значений тега	56
3.22 Выпадающий список	58
3.23 Поле ввода значений тега	60
3.24 Термометр	62
3.25 Спидометр	65
3.26 Графический индикатор значений тега	68
3.27 Графический индикатор интервальных значений тега.....	70
3.28 Линейный график	72
3.29 Флажок	77
3.30 Переключатель.....	79
3.31 Поле вывода ошибок	81
3.32 Журнал событий (log).....	84
3.33 Сохранение значений тега в файл	87
4 Часто задаваемые вопросы	90



1 Введение

Библиотека «Jet Logic Web Framework» (далее «библиотека») содержит файлы с кодом на JavaScript и CSS и предназначена для разработки встраиваемого web-сайта в устройства связи PL302, PL307 и другие совместимые. Web-сайт позволяет реализовать графический интерфейс управления различными устройствами и может запускаться на различных устройствах при помощи браузера.

Библиотека использует SVG графику для отображения компонентов управления, что позволяет динамически менять масштаб для каждого конкретного устройства без потери качества изображения.

Для взаимодействия с устройством, библиотека использует AJAX. Все необходимые запросы, ограничения запросов и их оптимизация выполняется автоматически. Управление процессом работы с AJAX контролируется специальными глобальными переменными (см. п. 2.4).



2 Использование библиотеки

2.1 Установка

Для использования библиотеки необходимо включить в состав встраиваемого сайта два файла: «pl.min.js» и «pl.min.css». Код встраиваемого сайта в свою очередь размещается в папке «/web» на SD-карте, устанавливаемой в устройство связи.

2.2 Минимальный шаблон страницы HTML

Для построения графической схемы необходимо применить следующий минимальный шаблон страницы HTML:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <link rel="stylesheet" type="text/css" href="pl.min.css">
  </head>
  <body>
    <div id="pl-gui" width=800 height=800>
      <!-- Место для добавления компонентов -->
    </div>
    <script src="pl.min.js"></script>
  </body>
</html>
```

Обязательным условием является наличие блока `<div id="pl-gui"></div>`. Компоненты графического интерфейса могут быть объявлены только внутри данного блока, учитывая степень вложенности (влияет на z-index).

Параметр `width` - задает ширину виртуальной области, которая автоматически масштабируется по ширине окна браузера на устройстве отображения. Благодаря этому независимо от реального разрешения экрана устройства отображается всё содержимое без необходимости прокрутки влево/вправо.

Параметр `height` необходим для сохранения соотношения сторон.

Все компоненты, находящиеся в блоке `<div id="pl-gui"></div>` используют только виртуальные координаты. Например, если указать `<div id="pl-gui" width=1000 height=500></div>` и поместить внутрь блока компонент с шириной 1000, то компонент займет всю ширину экрана (вне зависимости от устройства), 500 - половину ширины экрана и т.д. Виртуальные координаты (измеряются в виртуальных пикселях **vpх**) учитываются при всех операциях с координатами (размер шрифта, ширина, высота, ширина границы и т.п.). Такой подход гарантирует эквивалентное отображение графического интерфейса для всех поддерживаемых устройств.

Чтобы страница, оптимизированная под узкий экран смартфона, не выглядела чрезмерно большой на широком экране компьютера можно в блоке `id="pl-gui"` указать параметр «`max-width`». Например, так:

```
<div id="pl-gui" width=1080 height=1920 max-width="220mm" style="margin:
auto; background-color: white">
```

Параметр `style="margin: auto"` выравнивает страницу по центру окна браузера.

2.3 Связь с переменными во внешних устройствах

Чтобы связать переменные во внешних устройствах, подключаемых к PL302, с графическими объектами, объявляемыми в коде, необходимо их описать в «Таблице опроса переменных» используя веб-интерфейс настроек в PL302. Либо можно подготовить и записать на SD-карту в папку `sys\serv` файл `fileio.csv`.

Для входа в настройки набрать в адресной строке браузера «`http://ip-адрес/cfgPL/index.html`».

Модуль PL302 автоматически ведёт обмен данными с внешними устройствами и поддерживает в своей памяти актуальную базу текущих значений по всем переменным, описанным в таблице. А ядро библиотеки автоматически загружает в браузер текущие значения переменных, используемых в объектах на страницах, и наоборот, передаёт в PL302 значения переменных, изменённых пользователем. Каких-либо дополнительных действий от пользователя не требуется.

Связывание осуществляется через имена тегов, описанных в таблице опроса. Имена тегов указываются при объявлении соответствующих объектов. Формат описан в разделе «Графические компоненты».

Библиотека поддерживает управление настройками связи используя глобальные переменные (`DEFAULT_TIMEOUT`, `AJAX_TIMEOUT`, `AJAX_RETRIES`, `MAX_GET_TAGS_COUNT`, `MAX_SET_TAGS_COUNT`).

2.4 Глобальные переменные

Глобальные переменные позволяют оптимальным образом настроить поведение.

Список глобальных переменных с их описанием приведен в таблице ниже.

Блок с объявлением глобальных переменных должен быть расположен перед включением скрипта библиотеки JetLogic Web Framework.

Пример объявления глобальных переменных:

```
<script>
    HIDE_ERROR_DIALOG = true;
    AJAX_TIMEOUT = 1500;
    AJAX_RETRIES = 5;
    MAX_GET_TAGS_COUNT = 8;
    MAX_SET_TAGS_COUNT = 2;
</script>
<script src="pl.min.js"></script>
```

Таблица - глобальные переменные AJAX

Наименование	Значение по-умолч.	Ед. измер.	Описание
<code>HIDE_ERROR_DIALOG</code>	<code>False</code>	<code>bool</code>	Скрывать модальное диалоговое окно ошибки работы с тегами устройства
<code>DEFAULT_TIMEOUT</code>	<code>1000</code>	<code>мс</code>	Период опроса тегов

AJAX_TIMEOUT	1000	мс	Максимальное время ожидания ответа по AJAX
AJAX_RETRIES	3	шт.	Максимальное количество попыток запроса тега
MAX_GET_TAGS_COUNT	10	шт.	Максимальное количество тегов, отправляемых в запросе на получение значений тега
MAX_SET_TAGS_COUNT	4	шт.	Максимальное количество тегов, отправляемых в запросе на установку значений тега

Таблица - глобальные переменные графического интерфейса

Наименование	Значение по-умолч.	Ед. измер.	Описание
RX	0	vpx	Горизонтальный радиус скругления углов по умолчанию у всех компонентов (по умолчанию скругления нет).
RY	0	vpx	Вертикальный радиус скругления углов по умолчанию у всех компонентов (по умолчанию скругления нет).
FILL	transparent	HTML color	Цвет компонента по умолчанию.
STROKE	black	HTML color	Цвет границы по умолчанию.
STROKE_WIDTH	1	vpx	Толщина границы по умолчанию.
FONT_COLOR	black	HTML color	Цвет шрифта по умолчанию.
FONT_FAMILY	Arial	строка	Шрифт текста по умолчанию.
FONT_SIZE	10	vpx	Размер шрифта по умолчанию.
TEXT_ANCHOR	middle	Start, middle, end, inherit	Выравнивание текста по горизонтали по умолчанию.
ALIGNMENT_BASELINE	middle	auto baseline, before-edge, text-before-edge, middle, central, after-edge, text-after-edge,	Выравнивание текста по вертикали по умолчанию.

		ideographic, alphabetic, hanging, mathematical, inherit	
ERROR_FILL	rgba(0,0,0, 0.5)	HTML color	Цвет, накладываемый на компонент, когда возникают проблемы в получении данных с устройства.

2.5 Создание объектов из кода JS

Библиотека позволяет динамическое создание компонентов управления из javascript кода. Скрипт создания компонентов обязательно должен быть объявлен после подключения библиотеки PL302. Создаваемые элементы оперируют виртуальными координатами (см. п.2.2). Подробное описание того, как создавать компоненты из JS кода подробно описаны в соответствующих разделах (см. п.3).

Пример создания компонента Line из JS кода:

```
var line = PL.GUI.createLine({x1: 400, y1: 200, x2: 800, y2: 200});
```

2.6 Атрибуты отображения объектов

Библиотека использует масштабируемую SVG графику (позволяет отображать объекты без потери качества).

Для компонентов библиотеки используются только атрибуты, указанные в описании (п.3). Такой подход объясняется не только необходимостью инвариантного отображения элементов для всех доступных браузеров, но и тем, что часть компонентов представляют собой составной объект, изменение подэлемента которого приведет к некорректному поведению (например, изменение высоты шрифта для шкалы компонента термометра приведет к наложению текста).

Пользователь также может объявлять любые другие теги (HTML, SVG) в блоке **pl-gui**. В этом случае библиотека самостоятельно изменяет размер, положение и z-index (индекс положения на оси z, см. спецификацию W3C) для корректного отображения пользовательского интерфейса.

2.7 Событийная модель воздействий

Библиотека позволяет использовать обработку событий HTML. При создании компонента в JS возвращается презентер компонента. Презентер компонента обеспечивает привязку обработчиков событий, используя функцию `bindEvent(eventName, function)`. Ключевой особенностью является вызов обработчика событий под замыканием самого презентера, что упрощает программирование.

Пример:

```
var clickLine = PL.GUI.createLine({x1: 400,y1: 200, x2: 800, y2: 200,
```



```
stroke: 'blue', 'stroke-width': 100});  
clickLine.bindEvent('click', function(){  
  this.setState({  
    stroke: this.getState().stroke === 'blue' ? '#cccccc' : 'blue'  
  });  
});
```

Для использования обработчиков событий в описании HTML применяются 2 подхода. Первый заключается в использовании тега `events`, внутри которого помещается JSON объект (для экранирования кавычек необходимо использовать `\\`) следующего содержания:

```
<div class="line" x1="0" x2="400" y1="200" y2="200" stroke="red" stroke-  
width="100" events='{"click": "this.setState({stroke:  
this.getState().stroke === \\\\"red\\" ? \\\\"#cccccc\\" :  
\\\\"red\\"});"}'></div>
```

Для упрощения работы возможно использовать прямые теги для обработчиков событий (в обработчик также передается замыкание для презентера компонента), например:

```
<div class="line" x1="0" x2="100" y1="300" y2="300" stroke="red" stroke-  
width="50" onclick="alert(this);"></div>
```

Если возникает необходимость обрабатывать значения тегов вручную (декорирование), можно воспользоваться специальным событием **value**:

```
var element = PL.GUI.getComponentById('element');  
element.bindEvent('value', function(name, value){  
  console.log(name, value);  
});
```

2.8 Динамическое изменение атрибутов компонента

Для динамического изменения атрибутов устройства используется функция `setState(state)` для объекта презентера компонента (включая замыкание внутри обработчиков событий). Получение презентера объекта по `id` возможно при помощи `PL.GUI.getComponentById(id)`;

```
var clickLine = PL.GUI.createLine({x1: 400, y1: 200, x2: 800, y2: 200,  
stroke: 'blue', 'stroke-width': 100});  
clickLine.setState({  
  stroke: clickLine.getState().stroke === 'blue' ? '#cccccc' : 'blue'  
});
```




3 Графические компоненты

3.1 Окружность

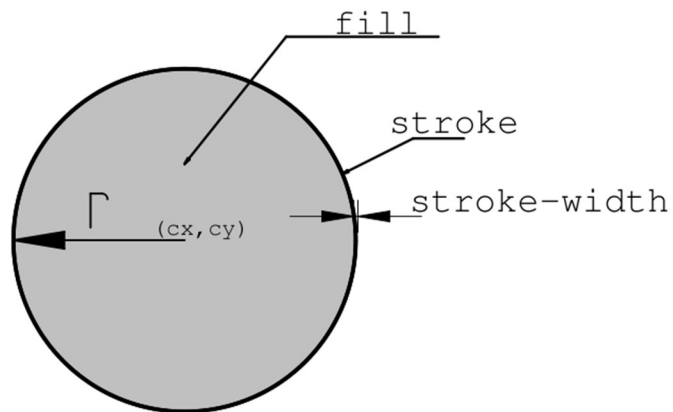


Рисунок 3.1 - Окружность

Обеспечивает отрисовку окружности (Рис. 3.1) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.1 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
cx	Число	0	Абсцисса центра окружности
cy	Число	0	Ордината центра окружности
r	Число	0	Радиус окружности

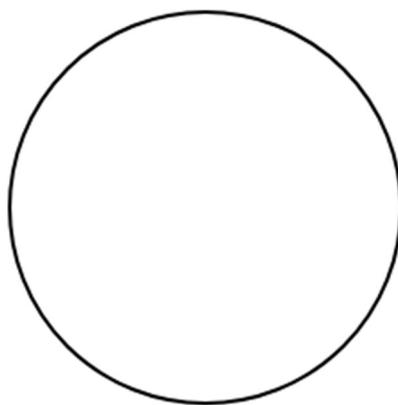


Рисунок 3.2 - Окружность с обязательными параметрами

Таблица 3.2 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="circle" cx="100" cy="100" r="40" ></div></code>

JavaScript `PL.GUI.createCircle({cx: 100 , cy: 100, r: 40})`

Таблица 3.3 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет окружности
stroke	Строка	black	Цвет границы окружности
stroke-width	Число	1	Толщина границы окружности

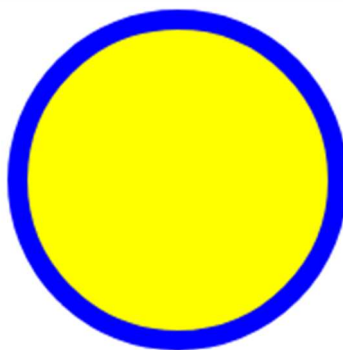


Рисунок 3.3 - Окружность с дополнительными параметрами

Таблица 3.4 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<code><div class="circle" cx="530" cy="320" r="40" stroke-width="5" stroke="blue" fill="yellow"></div></code>
JavaScript	<code>PL.GUI.createCircle({cx: 710, cy: 320, r: 40, 'stroke-width': 5, stroke: 'blue', fill: 'yellow'});</code>



3.2 Эллипс

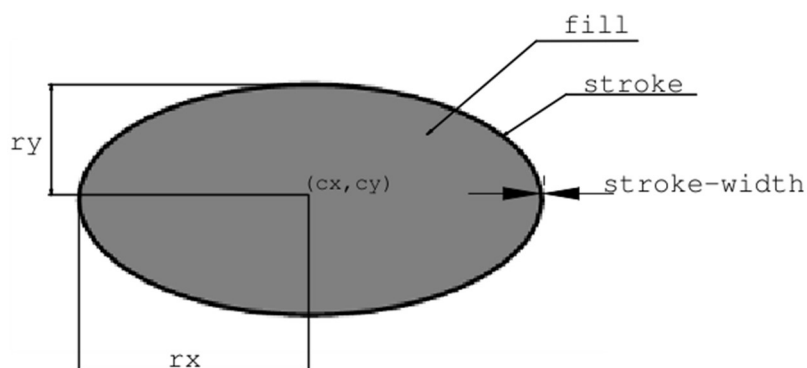


Рисунок 3.4 – Эллипс

Обеспечивает отрисовку эллипса (Рис. 3.4) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.5 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
cx	Число	0	Абсцисса центра эллипса
cy	Число	0	Ордината центра эллипса
rx	Число	0	Абсцисса радиуса эллипса
ry	Число	0	Ордината радиуса эллипса

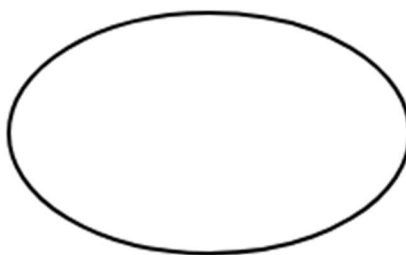


Рисунок 3.5 - Эллипс с обязательными параметрами

Таблица 3.6 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="ellipse" cx="100" cy="100" r="40" rx="50" ry="5"></div></code>
JavaScript	<code>PL.GUI.createEllipse({cx: 100, cy: 100, rx: 50, ry: 5, r: 40});</code>



Таблица 3.7 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет эллипса
stroke	Число	black	Цвет границы эллипса
stroke-width	Число	1	Толщина границы эллипса



Рисунок 3.6 - Эллипс с дополнительными параметрами

Таблица 3.8 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<code><div class="ellipse" cx="610" cy="250" rx="50" ry="10" fill="blue" stroke="yellow" stroke-width="1"></div></code>
JavaScript	<code>PL.GUI.createEllipse({cx: 610, cy: 250, rx: 50, ry: 10, fill: "blue", 'stroke-width': 1, stroke: "yellow"});</code>



3.3 Прямоугольник

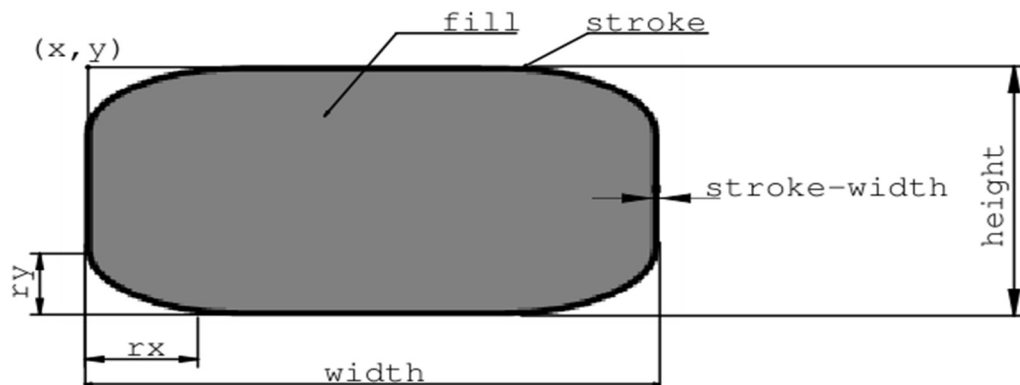


Рисунок 3.7 – Прямоугольник

Обеспечивает отрисовку прямоугольника (Рис. 3.7) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.9 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса прямоугольника
y	Число	0	Ордината прямоугольника
width	Число	0	Ширина прямоугольника
height	Число	0	Высота прямоугольника



Рисунок 3.8 - Прямоугольник с обязательными параметрами

Таблица 3.10 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="rect" x="100" y="100" width="100" height="50"></div></code>
JavaScript	<code>PL.GUI.createRect({x:100, y:100, width: 100, height: 50});</code>

Таблица 3.11 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет прямоугольника
stroke	Строка	black	Цвет границы прямоугольника



stroke-width	Число	1	Толщина границы прямоугольника
rx	Число	0	Горизонтальный радиус скругления углов прямоугольника
ry	Число	0	Вертикальный радиус скругления углов прямоугольника

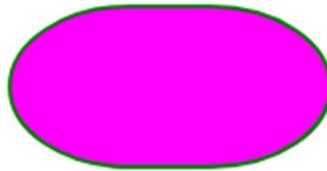


Рисунок 3.9 - Прямоугольник с дополнительными параметрами

Таблица 3.12 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="rect" x="610" y="150" width="80" height="40" stroke-width="1" fill="magenta" stroke="green" rx="30" ry="30"></div></pre>
JavaScript	<pre>PL.GUI.createRect({x:610, y:150, width: 80, height: 40, 'stroke-width': 1, fill: "magenta", stroke: "green", rx: 30, ry: 30});</pre>

3.4 Линия

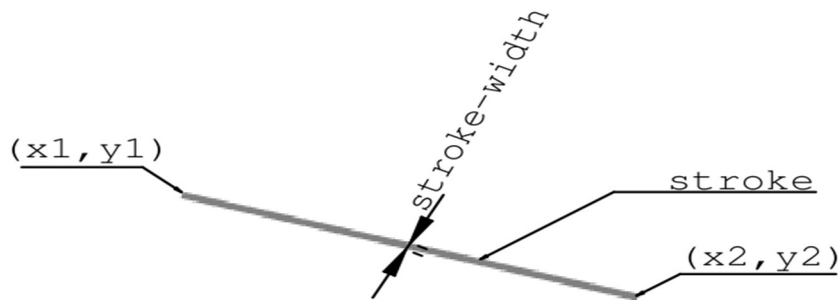


Рисунок 3.10 - Линия

Обеспечивает отрисовку линии (Рис. 3.10) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.13 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x1	Число	0	Координата начала линии по оси абсцисс
y1	Число	0	Координата начала линии по оси ординат
x2	Число	0	Координата конца линии по оси абсцисс
y2	Число	0	Координата конца линии по оси ординат

Рисунок 3.11 - Линия с обязательными параметрами

Таблица 3.14 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="line" x1="100" y1="100" x2="250" y2="300"></div></code>
JavaScript	<code>PL.GUI.createLine({x1: 100, y1: 100, x2: 250, y2: 300});</code>

Таблица 3.15 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
stroke	Строка	black	Цвет линии
stroke-width	Число	1	Толщина линии

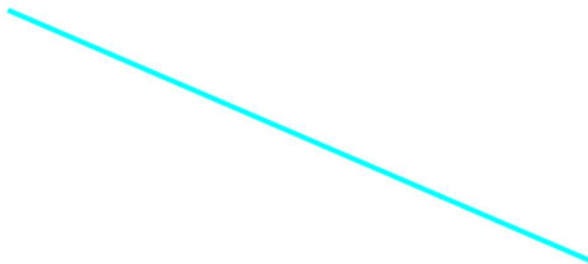


Рисунок 3.12 - Линия с дополнительными параметрами

Таблица 3.16 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<code><div class="line" x1="400" x2="500" y1="10" y2="10" stroke="lightblue" stroke-width="5"></div></code>
JavaScript	<code>PL.GUI.createLine({x1: 400, y1: 10, x2: 500, y2: 10, stroke: 'lightblue', 'stroke-width': 2});</code>

3.5 Ломанная

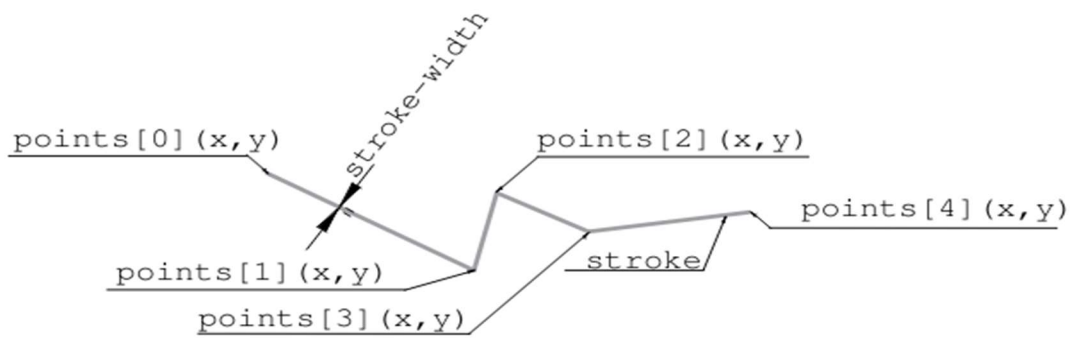


Рисунок 3.13 - Ломанная

Обеспечивает отрисовку ломанной (Рис. 3.13) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.17 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
<code>points</code>	Массив	<code>[0, 0]</code>	Массив двумерных точек для отрисовки



Рисунок 3.14- Ломанная с обязательными параметрами

Таблица 3.18 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<pre><div class="polyline" points="[100,100,250,200,300,150,350,250,500,100]"> </div></pre>
JavaScript	<pre>PL.GUI.createPolyline({points:[100,100,250,200,300,150,350,250,500,100]});</pre>

Таблица 3.19 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
<code>stroke</code>	Строка	<code>black</code>	Цвет ломанной
<code>stroke-width</code>	Число	<code>1</code>	Толщина ломанной



Рисунок 3.15- Ломанная с дополнительными параметрами

Таблица 3.20 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="polyline" points="[100,100,250,200,300,150,350,250,500,100]" stroke="lightblue" stroke-width="2"></div></pre>
JavaScript	<pre>PL.GUI.createPolyline({points:[100,100,250,200,300,150,350,250,500,100], stroke: 'lightblue', 'stroke-width': 2});</pre>

3.6 Полигон

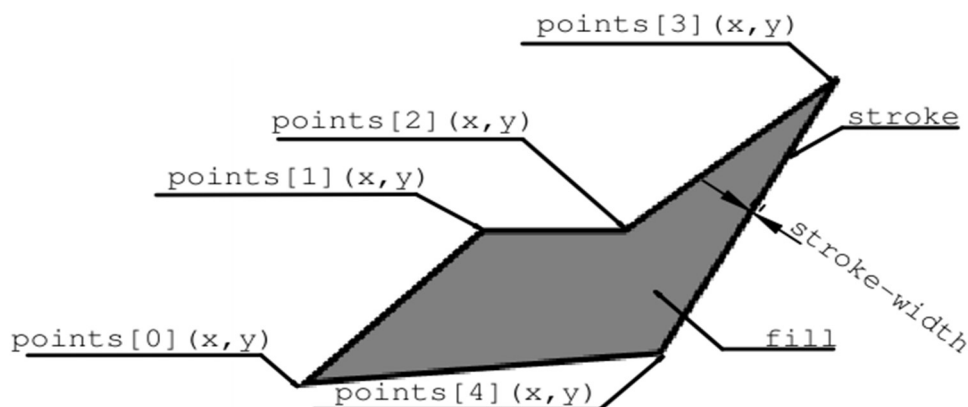


Рисунок 3.16 - Полигон

Обеспечивает отрисовку полигона (Рис. 3.16) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.21 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
points	Массив	[0,0]	Массив двумерных точек для отрисовки полигона

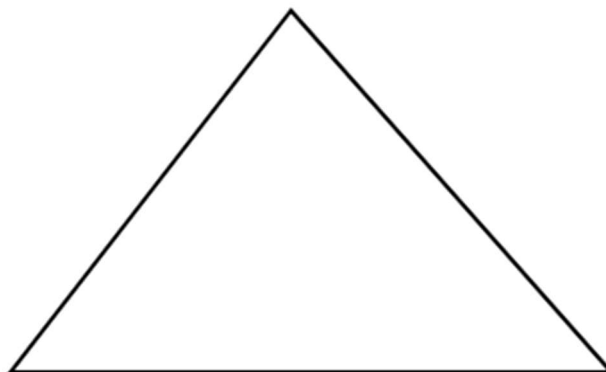


Рисунок 3.17 - Полигон с обязательными параметрами

Таблица 3.22 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<pre><div class="polygon" points="[100,150,200,140,250,50,190,100,150,100]"> </div></pre>
JavaScript	<pre>PL.GUI.createPolygon({points: [100,150,200,140,250,50,190,100,150,100]});</pre>

Таблица 3.23 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет полигона
stroke	Строка	black	Цвет границы полигона



stroke-width	Число	1	Толщина границы полигона
--------------	-------	---	--------------------------

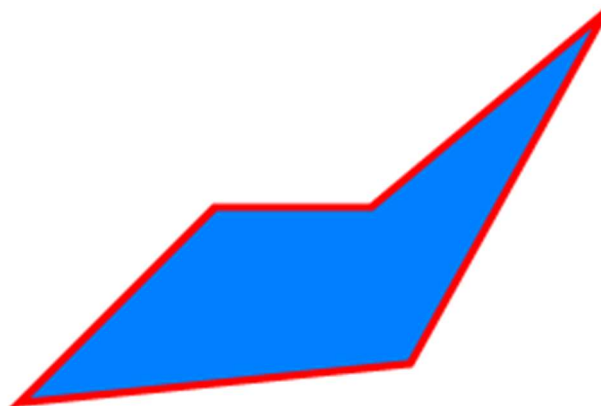


Рисунок 3.18 - Полигон с дополнительными параметрами

Таблица 3.24 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="polygon" points="[100,150,200,140,250,50,190,100,150,100]" stroke="red" stroke-width="2" fill="blue"></div></pre>
JavaScript	<pre>PL.GUI.createPolygon({points: [100,150,200,140,250,50,190,100,150,100], stroke: 'red', 'stroke-width': 2, fill: 'blue'});</pre>

3.7 Текст

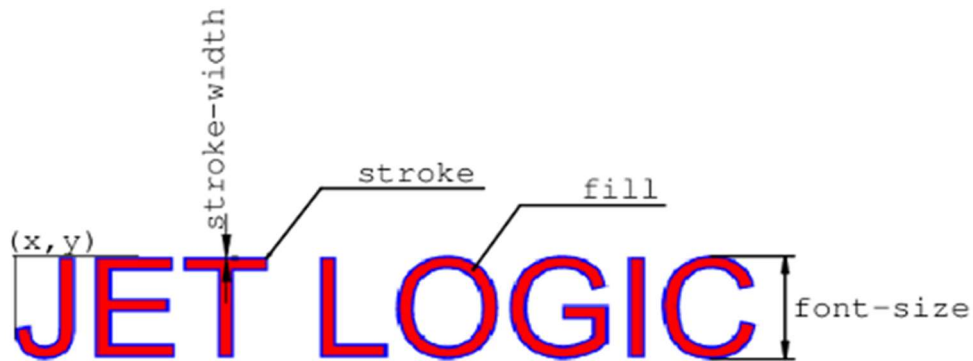


Рисунок 3.19 - Текст

Обеспечивает отрисовку текста (Рис. 3.19) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.25 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса точки привязки текста
y	Число	0	Ордината точки привязки текста
text	Строка	""	Текст надписи



Рисунок 3.20 - Текст с обязательными параметрами

Таблица 3.26 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="text" x="200" y="300" text="JET LOGIC"></div></code>
JavaScript	<code>PL.GUI.createText({x:200, y: 300, text: 'JET LOGIC'});</code>

Таблица 3.27 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	black	Цвет текста
font-size	Число	14	Размер текста
font-family	Строка	Arial	Шрифт текста
stroke	Строка	black	Цвет границы текста
stroke-width	Число	0	Толщина границы текста



text-anchor	Строка	middle	Горизонтальное центрирование текста
alignment-baseline	Строка	middle	Вертикальное центрирование текста

JET LOGIC

Рисунок 3.21 - Текст с дополнительными параметрами

Таблица 3.28 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="text" x="200" y="300" text="JET LOGIC" fill="red" font-size="40" font-family="Arial" stroke="blue" stroke-width="3" text-anchor="start" alignment-baseline="hanging"></div></pre>
JavaScript	<pre>PL.GUI.createText({x:200, y: 300, text: 'JET LOGIC', fill: 'red', 'font-size': 40, 'font-family': 'Arial', stroke: 'blue', 'stroke-width': 3, 'text-anchor': 'start', 'alignment-baseline': 'hanging'});</pre>

3.8 Изображение

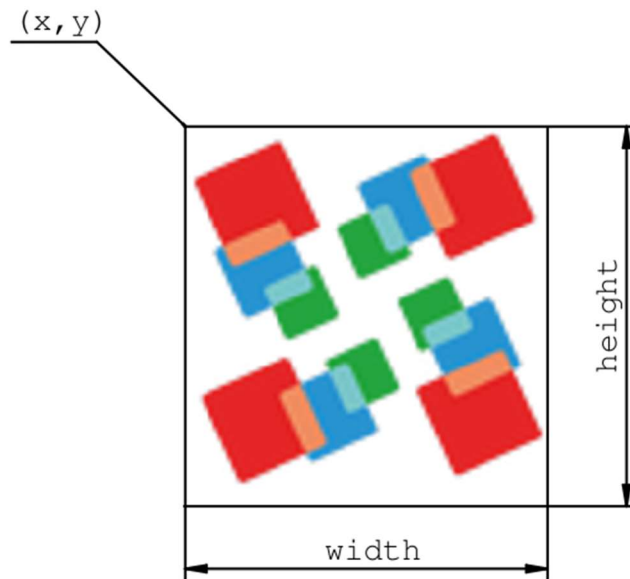


Рисунок 3.22 - Изображение

Обеспечивает отрисовку изображения (Рис. 3.22) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.29 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла изображения
y	Число	0	Ордината верхнего левого угла изображения
xlink:href (src)	Строка	undefined	URL изображения (путь изображения)
width	Число	0	Ширина изображения
height	Число	0	Высота изображения



Рисунок 3.23 – Изображение с параметрами

Таблица 3.30 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="image" x="100" y="100" xlink:href="logo.png" width="100" height="100"></div></code>
JavaScript	<code>PL.GUI.createImage({x: 100, y:100, 'xlink:href': "logo.png", width: 100, height: 100});</code>

Таблица 3.31 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
preserve-aspect-ratio	Строка	none	Масштабирование изображения. Возможные варианты: xMinYMin xMidYMin xMaxYMin xMinYMid xMidYMid xMaxYMid xMinYMax xMidYMax xMaxYMax

Таблица 3.32 – Preserve Aspect Ratio

Параметр	Описание
xMin	Выравнивание в соответствии с минимальным значением x viewBox – левая граница области просмотра
xMid	Выравнивание в соответствии с центральной точкой по x viewBox – центр по оси X в области просмотра
xMax	Выравнивание в соответствии с максимальным значением x viewBox – правая граница области просмотра
YMin	выровнять в соответствии с минимальным значением y viewBox – верхняя граница области просмотра
YMid	выровнять в соответствии с центральной точкой по y viewBox – центр по оси Y в области просмотра
YMax	выровнять в соответствии с максимальным значением y viewBox – нижняя граница области просмотра

Таблица 3.33 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<code><div class="image" x="100" y="100" xlink:href="logo.png" width="100" height="100" preserve-aspect-ratio="xMaxYMin"></div></code>
JavaScript	<code>PL.GUI.createImage({x: 100, y:100, 'xlink:href': 'logo.png', width: 100, height: 100 'preserve-aspect-ratio': 'xMaxYMin'});</code>

3.9 Графический индикатор и переключатель битовых состояний

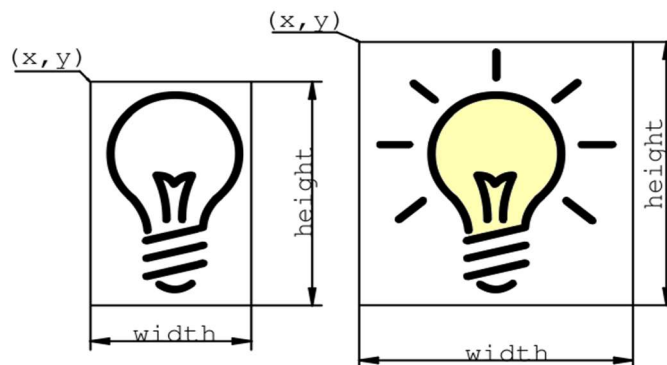


Рисунок 3.24 - Графический индикатор и переключатель битовых состояний

Обеспечивает отрисовку графического индикатора и переключателя битовых состояний (Рис. 3.24) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.34 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
true-image	Строка	undefined	URL истинного изображения
false-image	Строка	undefined	URL ложного изображения
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина графического индикатора и переключателя битовых состояний
height	Число	0	Высота графического индикатора и переключателя битовых состояний
name	Строка	undefined	Имя тега
bit-number	Число	undefined	Номер бита

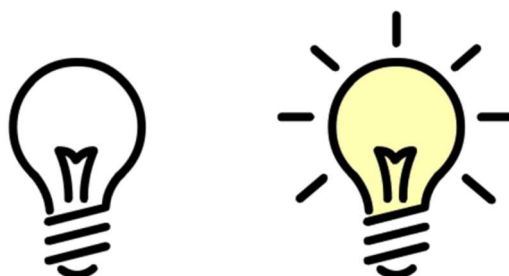


Рисунок 3.25 – Графический индикатор с параметрами

Таблица 3.35 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<pre><div class="bool-image" x=100 y=100 width=256 height=256 true-image="true.png" false-image="false.png" bit-number="1" name="tag1"></div></pre>

```
JavaScript PL.GUI.createBoolImage({x: 100, y: 100, width: 256,
height: 256, 'true-image': "true.png", 'false-image':
"false.png", name: "tag1", 'bit-number': 1});
```

Таблица 3.36 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
preserve-aspect-ratio	Строка	none	Масштабирование изображения. Возможные варианты: xMinYMin xMidYMin xMaxYMin xMinYMid xMidYMid xMaxYMid xMinYMax xMidYMax xMaxYMax
error-fill	Строка	rgba(0,0,0,0.5)	Цвет графического индикатора при ошибке
timeout	Число	1000	Период обновления (мс)

Таблица 3.37 – Preserve Aspect Ratio

Параметр	Описание
xMin	Выравнивание в соответствии с минимальным значением x viewBox – левая граница области просмотра
xMid	Выравнивание в соответствии с центральной точкой по x viewBox – центр по оси X в области просмотра
xMax	Выравнивание в соответствии с максимальным значением x viewBox – правая граница области просмотра
YMin	Выравнивание в соответствии с минимальным значением y viewBox – верхняя граница области просмотра
YMid	Выравнивание в соответствии с центральной точкой по y viewBox – центр по оси Y в области просмотра
YMax	Выравнивание в соответствии с максимальным значением y viewBox – нижняя граница области просмотра

Таблица 3.38 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="bool-image" x=100 y=100 width=256 height=256 true-image="true.png" false- image="false.png" preserve-aspect-ratio="xMaxYMin" error-fill="rgba(255, 0, 0, 1)"></div></pre>
JavaScript	<pre>PL.GUI.createBoolImage({x: 100, y: 100, width: 256, height: 256, 'true-image': "true.png", 'false-image': "false.png", 'preserve-aspect-ratio': 'xMaxYMin', 'error-fill': 'rgba(255, 0, 0, 1)'});</pre>



3.10 Слайдер

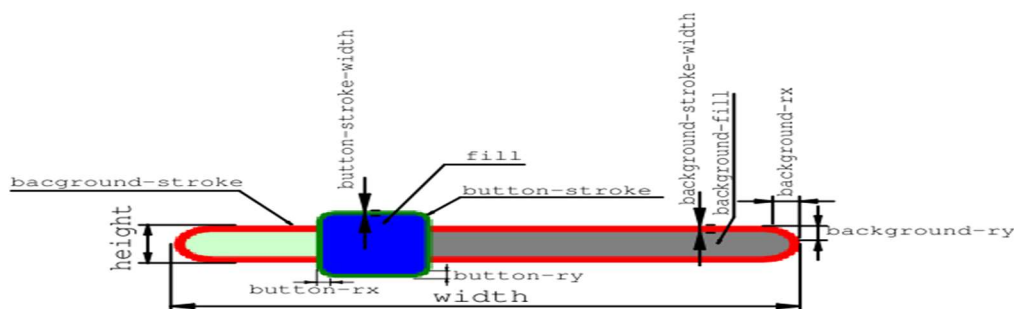


Рисунок 3.26 - Слайдер

Обеспечивает отрисовку слайдера (Рис. 3.26) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.39 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина слайдера
height	Число	0	Высота слайдера
min-value	Число	0	Минимальное значение тега
max-value	Число	10000	Максимальное значение тега
name	Строка	undefined	Имя тега

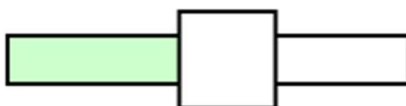


Рисунок 3.27 – Слайдер с обязательными параметрами

Таблица 3.40 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="slider" x="100" y="100" width="170" height="15" min-value="500" max-value="5000" name="tag1"></div></code>
JavaScript	<code>PL.GUI.createSlider({x:100, y: 100, width: 170, height: 15, 'min-value': 500, 'max-value': 5000, name: "tag1"});</code>

Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
----------	--------------	-----------------------	----------



background-rx	Число	0	Горизонтальный радиус скругления углов полосы состояния
background-ry	Число	0	Вертикальный радиус скругления углов полосы состояния
button-rx	Число	0	Горизонтальный радиус скругления углов ползунка
button-ry	Число	0	Вертикальный радиус скругления углов ползунка
background-stroke	Строка	black	Цвет границы полосы состояния
button-stroke	Строка	black	Цвет границы ползунка
button-stroke-width	Число	1	Ширина границы ползунка
background-stroke-width	Число	1	Ширина границы полосы состояния
data-colors	Массив	[]	Цветовой индикатор данных
background-fill	Строка	transparent	Цвет слайдера
button-fill	Строка	white	Цвет ползунка
error-fill	Строка	rgba(0,0,0,0.5)	Цвет слайдера при ошибке
timeout	Число	1000	Период обновления (мс)

Рисунок 3.28 – Слайдер с дополнительными параметрами

Таблица 3.41 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="slider" x="30" y="180" name="tag1" width="50" height="10" min-value="500" max- value="5000" button-fill="red" button- stroke="magenta" button-stroke-width="4" button- rx="4" button-ry="4" background-rx="4" background- ry="4" background-stroke="blue" button-stroke="black" background-stroke-width="2" error-fill="rgba(255, 0, 0, 1) data-colors='{ "rgba(255,0,0)": [0,1500] }' ></div></pre>
JavaScript	<pre>PL.GUI.createSlider({x:30, y: 180, width: 50, height: 10, 'min-value': 500, 'max-value': 5000, name: "tag1", 'button-fill': 'red', 'button-stroke': 'magenta', 'button-stroke-width': 4, 'button-rx': 4, 'button-ry': 4, 'background-rx': 4, 'background-ry': 4, 'background-stroke': 'blue', 'button-stroke': 'black', background-stroke-width': '2', 'error-fill': 'rgba(255, 0, 0, 1)', 'data-colors': '{"&quot;rgba(255,0,0)&quot;:[0,1500] }'});</pre>

3.11 Вертикальный слайдер

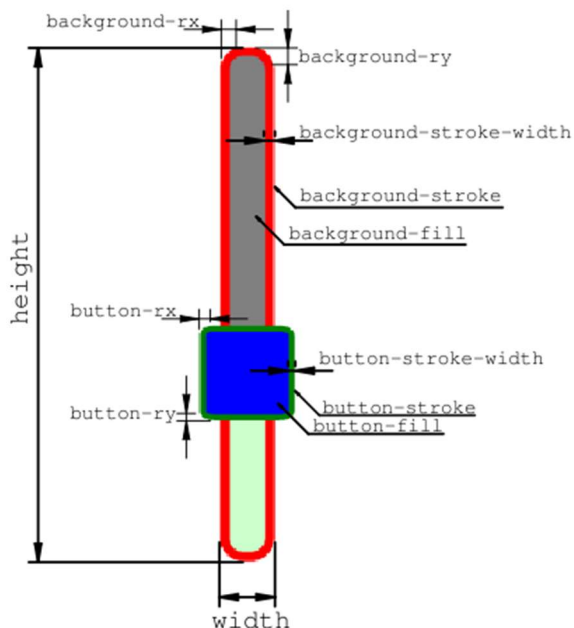


Рисунок 3.29 - Вертикальный слайдер

Обеспечивает отрисовку вертикального слайдера (Рис. 3.29) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.42 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина слайдера
height	Число	0	Высота слайдера
min-value	Число	0	Минимальное значение тега
max-value	Число	10000	Максимальное значение тега
name	Строка	undefined	Имя тега

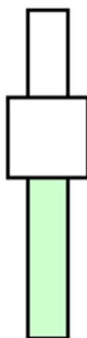


Рисунок 3.30 – Вертикальный слайдер с обязательными параметрами

Таблица 3.43 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="vertical-slider" x="100" y="100" width="15" height="170" min-value="500" max-value="5000" name="tag1"></div></code>
JavaScript	<code>PL.GUI.createVerticalSlider({x:100, y: 100, width: 15, height: 170, 'min-value': 500, 'max-value': 5000, name: "tag1"});</code>

Таблица 3.44 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
background-rx	Число	0	Горизонтальный радиус скругления углов полосы состояния
background-ry	Число	0	Вертикальный радиус скругления углов полосы состояния
button-rx	Число	0	Горизонтальный радиус скругления углов ползунка
button-ry	Число	0	Вертикальный радиус скругления углов полосы ползунка
button-fill	Строка	white	Цвет ползунка
background-fill	Строка	transparent	Цвет полосы состояния
background-stroke	Строка	black	Цвет границы полосы состояния
button-stroke	Строка	black	Цвет границы ползунка
button-stroke-width	Число	1	Ширина границы ползунка
background-stroke-width	Число	1	Ширина границы полосы состояния
data-colors	Массив	[]	Цветовой индикатор данных
error-fill	Строка	rgba(0,0,0,0.5)	Цвет вертикального слайдера при ошибке
timeout	Число	1000	Период обновления (мс)

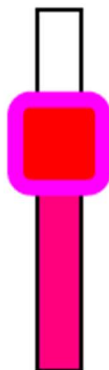


Рисунок 3.31 – Вертикальный слайдер с дополнительными параметрами

Таблица 3.45 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="vertical-slider" x="30" y="180" name="tag1" width="50" height="10" min-value="500" max-value="5000" button-fill="red" button- stroke="magenta" button-stroke-width="4" button- rx="4" button-ry="4" background-rx="4" background- ry="4" background-stroke="blue" button-stroke="black" background-stroke-width="2" error-fill="rgba(255, 0, 0, 1) data- colors='{"rgba(255,0,0)": [0,1500] }'></div></pre>
JavaScript	<pre>PL.GUI.createVerticalSlider({x:30, y: 180, width: 50, height: 10, 'min-value': 500, 'max-value': 5000, name: "tag1", 'button-fill': 'red', 'button-stroke': 'magenta', 'button-stroke-width': 4, 'button-rx': 4, 'button-ry': 4, 'background-rx': 4, 'background-ry': 4, 'background-stroke': 'blue', 'button-stroke': 'black', background-stroke-width': '2', 'error-fill': 'rgba(255, 0, 0, 1)', 'data-colors': '{"&quot;rgba(255,0,0)&quot;:[0,1500] }"});</pre>



3.12 Полоса состояния

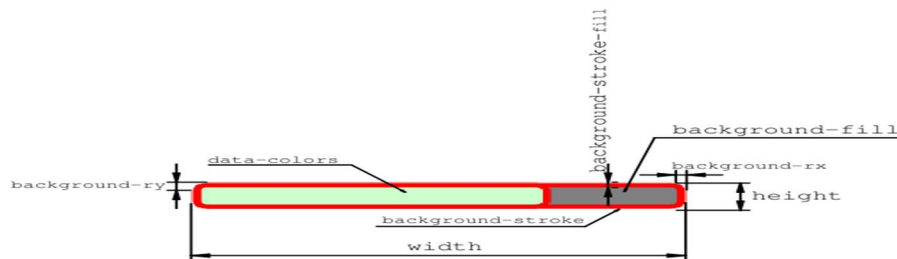


Рисунок 3.32 - Полоса состояния

Обеспечивает отрисовку полосы состояния (Рис. 3.32) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.46 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина полосы состояния
height	Число	0	Высота полосы состояния
min-value	Число	0	Минимальное значение тега
max-value	Число	10000	Максимальное значение тега
name	Строка	undefined	Имя тега

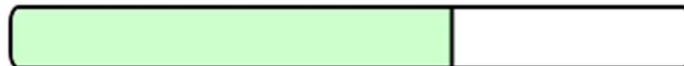


Рисунок 3.33 - Полоса состояния с обязательными параметрами

Таблица 3.47 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<pre><div class="bar" x="100" y="100" width="150" height="15" min-value="500" max-value="5000" name="tag1"></div></pre>
JavaScript	<pre>PL.GUI.createBar({x:100, y: 100, width: 150, height: 15, 'min-value': 500, 'max-value': 5000, name: 'tag1'});</pre>

Таблица 3.48 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
background-rx	Число	0	Горизонтальный радиус скругления углов полосы состояния
background-ry	Число	0	Вертикальный радиус скругления углов полосы



			состояния
background-fill	Строка	transparent	Цвет полосы состояния
background-stroke	Строка	black	Цвет границы полосы состояния
background-stroke-width	Число	1	Ширина границы полосы состояния
data-colors	Массив	[]	Цветовой индикатор данных
error-fill	Строка	rgba(0,0,0,0.5)	Цвет полосы состояния при ошибке
timeout	Число	1000	Период обновления (мс)



Рисунок 3.34 - Полоса состояния с дополнительными параметрами

Таблица 3.49 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="bar" x="30" y="180" name="tag1" width="50" height="10" min-value="500" max-value="5000" background-rx="4" background-ry="4" background-stroke="blue" button-stroke="black" background-stroke-width="2" background-fill="green" error-fill="rgba(255, 0, 0, 1) data-colors='{"rgba(255,0,0)": [0,1500] }'></div></pre>
JavaScript	<pre>PL.GUI.createBar({x:30, y: 180, width: 50, height: 10, 'min-value': 500, 'max-value': 5000, name: "tag1", 'background-rx': 4, 'background-ry': 4, 'background-stroke': 'blue', 'background-fill': 'green', 'button-stroke': 'black', background-stroke-width': '2', 'error-fill': 'rgba(255, 0, 0, 1)', 'data-colors': "{&quot;rgba(255,0,0)&quot;:[0,1500] }"});</pre>



3.13 Вертикальная полоса состояния

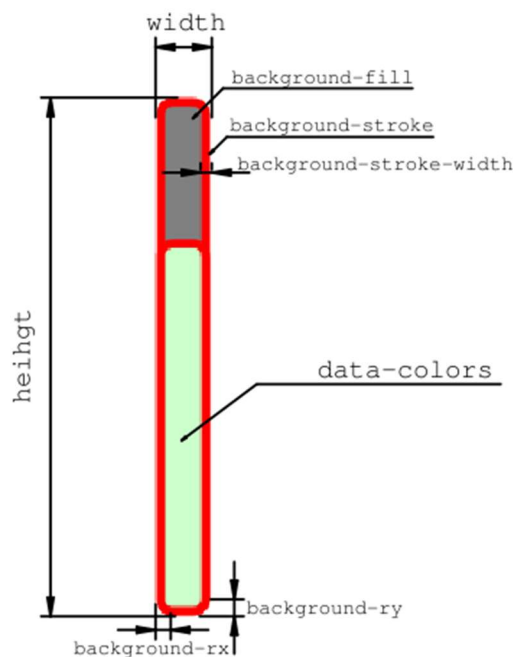


Рисунок 3.35 - Вертикальная полоса состояния

Обеспечивает отрисовку вертикальной полосы состояния (Рис. 3.35) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.50 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина полосы состояния
height	Число	0	Высота полосы состояния
min-value	Число	0	Минимальное значение тега
max-value	Число	10000	Максимальное значение тега
name	Строка	undefined	Имя тега



Рисунок 3.36 - Вертикальная полоса состояния с обязательными параметрами

Таблица 3.51 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="vertical-bar" x="100" y="100" width="15" height="150" min-value="500" max-value="5000" name="tag1"></div></code>
JavaScript	<code>PL.GUI.createVerticalBar({x:100, y: 100, width: 15, height: 150, 'min-value': 500, 'max-value': 5000, name: "tag1"});</code>

Таблица 3.52 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
background-rx	Число	0	Горизонтальный радиус скругления углов полосы состояния
background-ry	Число	0	Вертикальный радиус скругления углов полосы состояния
background-fill	Строка	transparent	Цвет полосы состояния
background-stroke	Строка	black	Цвет границы полосы состояния
background-stroke-width	Число	1	Ширина границы полосы состояния
data-colors	Массив	[]	Цветовой индикатор данных
error-fill	Строка	rgba(0,0,0,0.5)	Цвет вертикальной полосы состояния при ошибке
timeout	Число	1000	Период обновления (мс)



Рисунок 3.37 - Вертикальная полоса состояния с дополнительными параметрами

Таблица 3.53 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="vertical-bar" x="30" y="180" name="tag1" width="15" height="150" min-value="500" max- value="5000" background-rx="4" background-ry="4" background-stroke="blue" button-stroke="black" background-stroke-width="2" background-fill="green" error-fill="rgba(255, 0, 0, 1) data- colors='{ "rgba(255,0,0)": [0,1500] }'></div></pre>
JavaScript	<pre>PL.GUI.createVerticalBar({x:30, y: 180, width: 15, height: 150, 'min-value': 500, 'max-value': 5000, name: "tag1", 'background-rx': 4, 'background-ry': 4, 'background-stroke': 'blue', 'background-fill': 'green', 'button-stroke': 'black', background- stroke-width': '2', 'error-fill': 'rgba(255, 0, 0, 1)', 'data-colors': "{'&quot;rgba(255,0,0)&quot;:[0,1500] }"}");</pre>



3.14 Битовый переключатель

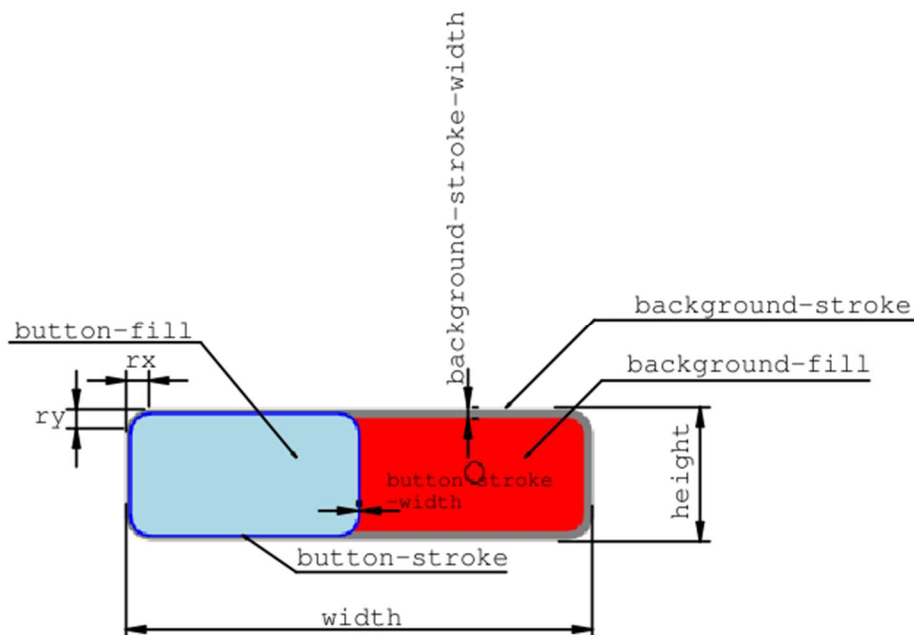


Рисунок 3.38 - Битовый переключатель

Обеспечивает отрисовку битового переключателя (Рис. 3.38) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.54 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина битового переключателя
height	Число	0	Высота битового переключателя
name	Строка	undefined	Имя тега
bit-number	Число	undefined	Номер бита

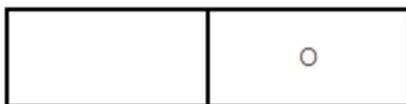


Рисунок 3.39 - Битовый переключатель с обязательными параметрами

Таблица 3.55 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<pre><div class="bool-slider" x="100" y="100" width="150" height="40" bit-number="1" name="tag1"></div></pre>
JavaScript	<pre>PL.GUI.createBoolSlider({x:100, y: 100, width: 150, height: 40, name: "tag1", 'bit-number': 1});</pre>

Таблица 3.56 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
rx	Число	0	Горизонтальный радиус скругления углов битового переключателя.
ry	Число	0	Вертикальный радиус скругления углов битового переключателя
background-fill	Строка	#ffcccc	Цвет неактивной зоны битового переключателя
button-fill	Строка	white	Цвет ползунка битового переключателя
active-fill	Строка	#ccffcc	Цвет активной зоны битового переключателя
background-stroke	Строка	black	Цвет границы битового переключателя.
button-stroke	Строка	black	Цвет границы ползунка битового переключателя
background-stroke-width	Число	1	Ширина границы битового переключателя
button-stroke-width	Число	1	Ширина ползунка битового переключателя
error-fill	Строка	rgba(0,0,0, 0.5)	Цвет битового переключателя при ошибке
timeout	Число	1000	Период обновления (мс)



Рисунок 3.40 - Битовый переключатель с дополнительными параметрами

Таблица 3.57 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<code><div class="bool-slider" x="100" y="100" width="150" height="40" bit-number="1" name="tag1" button-fill="red" button-stroke="magenta" button-stroke-width="4" rx="4" ry="4" background-stroke="blue" button-stroke="black" background-stroke-width="2" error-fill="rgba(255, 0, 0, 1)" ></div></code>
JavaScript	<code>PL.GUI.createBoolSlider({x:100, y: 100, width: 150, height: 40, name: "tag1", 'bit-number': 1, 'button-fill': 'red', 'button-stroke': 'magenta', 'button-stroke-width': 4, rx: 4, ry: 4, 'background-stroke': 'blue', 'button-stroke': 'black', background-stroke-width: '2', 'error-fill': 'rgba(255, 0, 0, 1)'}));</code>



3.15 Вертикальный битовый переключатель

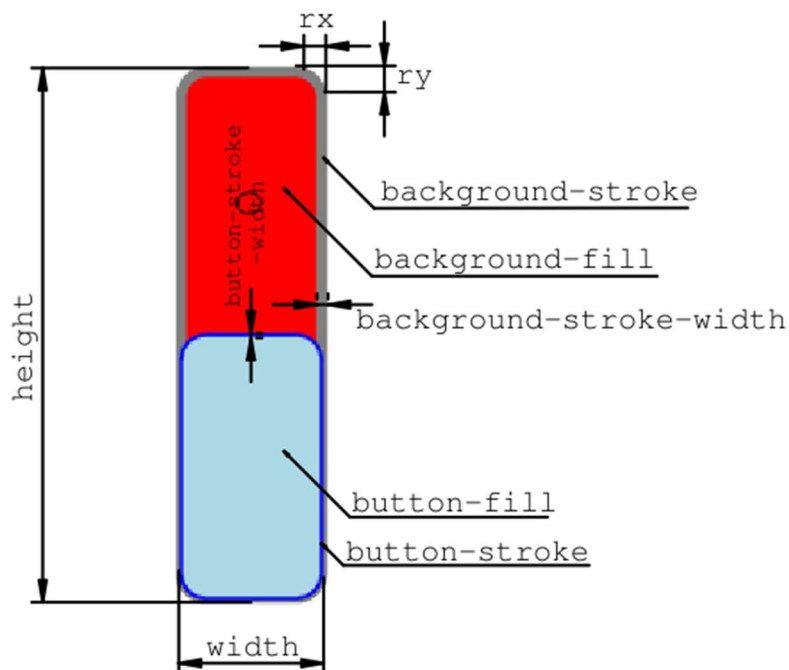


Рисунок 3.41 - Вертикальный битовый переключатель

Обеспечивает отрисовку вертикального битового переключателя (Рис. 3.41) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.58 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина вертикального битового переключателя
height	Число	0	Высота вертикального битового переключателя
name	Строка	undefined	Имя тега
bit-number	Число	undefined	Номер бита



Рисунок 3.42 - Вертикальный битовый переключатель с основными параметрами

Таблица 3.59 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="vertical-bool-slider" x="100" y="100" width="15" height="40" bit-number="1" name="tag1"></div></code>
JavaScript	<code>PL.GUI.createVerticalBoolSlider({x:100, y: 100, width: 15, height: 40, name: "tag1", 'bit-number': 1});</code>

Таблица 3.60 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
rx	Число	0	Горизонтальный радиус скругления углов вертикального битового переключателя
ry	Число	0	Вертикальный радиус скругления углов вертикального битового переключателя
background-fill	Строка	#ffcccc	Цвет неактивной зоны вертикального битового переключателя
button-fill	Строка	white	Цвет ползунка вертикального битового переключателя
active-fill	Строка	#ccffcc	Цвет активной зоны вертикального битового переключателя
background-stroke	Строка	black	Цвет границы вертикального битового переключателя
button-stroke	Строка	black	Цвет границы ползунка вертикального битового переключателя
background-stroke-width	Число	1	Ширина границы вертикального битового переключателя
button-stroke-width	Число	1	Ширина ползунка вертикального битового переключателя
error-fill	Строка	rgba(0,0,0, 0.5)	Цвет вертикального битового переключателя при ошибке
timeout	Число	1000	Период обновления (мс)

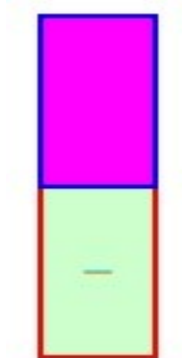


Рисунок 3.43 - Вертикальный битовый переключатель с дополнительными параметрами



Таблица 3.61 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="-vertical-bool-slider" x="100" y="100" width="150" height="40" bit-number="1" name="tag1" button-fill="red" button-stroke="magenta" button-stroke-width="4" rx="4" ry="4" background-stroke="blue" button-stroke="black" background-stroke-width="2" error-fill="rgba(255, 0, 0, 1)" ></div></pre>
JavaScript	<pre>PL.GUI.createVerticalBoolSlider({x:100, y: 100, width: 150, height: 40, name: "tag1", 'bit-number': 1, 'button-fill': 'red', 'button-stroke': 'magenta', 'button-stroke-width': 4, rx: 4, ry: 4, 'background-stroke': 'blue', 'button-stroke': 'black', 'background-stroke-width': '2', 'error-fill': 'rgba(255, 0, 0, 1)'}));</pre>

3.16 Кнопка

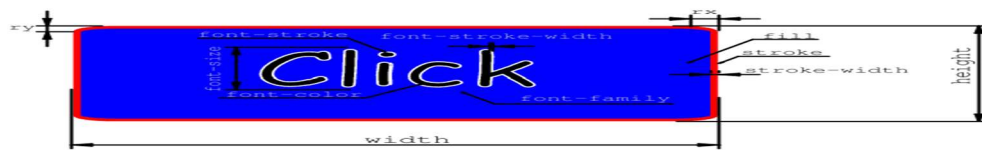


Рисунок 3.44 - Кнопка

Обеспечивает отрисовку кнопки (Рис. 3.44) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.62 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина кнопки
height	Число	0	Высота кнопки
text	Строка	Пустая строка	Текст кнопки



Рисунок 3.45 – Кнопка с обязательными параметрами

Таблица 3.63 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="button" x="100" y="100" width="40" height="150" text="Click"></div></code>
JavaScript	<code>PL.GUI.createButton({x:100, y: 100, width: 40, height: 150, text: 'Click'});</code>

Таблица 3.64 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
rx	Число	0	Горизонтальный радиус скругления углов кнопки
ry	Число	0	Вертикальный радиус скругления углов кнопки
fill	Строка	transparent	Цвет кнопки
stroke	Строка	black	Цвет границы кнопки
stroke-width	Число	1	Ширина границы кнопки
font-color	Строка	black	Цвет текста кнопки
font-size	Число	14	Размер текста кнопки



font-family	Строка	Arial	Шрифт текста кнопки
text-anchor	Строка	middle	Смещение текста по горизонтали
alignment-baseline	Строка	middle	Смещение текста по вертикали
font-stroke	Строка	black	Цвет границы текста кнопки
font-stroke-width	Число	0	Ширина границы текста кнопки



Рисунок 3.46 - Кнопка с дополнительными параметрами.

Таблица 3.65 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="button" x="100" y="100" width="40" height="150" text="Click" rx="15" ry="5" fill="#0000FF" stroke="#FF0000" stroke-width="3" font-color="#cccccc" font-size="40" font-family="Comic Sans MS" text-anchor="end" alignment-baseline="start" font-stroke="#FFFFFF" font-stroke-width="2" ></div></pre>
JavaScript	<pre>PL.GUI.createButton({x:100, y: 100, width: 40, height: 150, text: 'Click', rx: 15, ry: 5, fill: '#0000FF', stroke: '#FF0000', 'stroke-width': '3', 'font-color': '#cccccc', 'font-size': '40', 'font-family': 'Comic Sans MS', 'text-anchor': 'end', 'alignment-baseline': 'start', 'font-stroke': '#FFFFFF', 'font-stroke-width': '2'});</pre>

3.17 Графическая кнопка

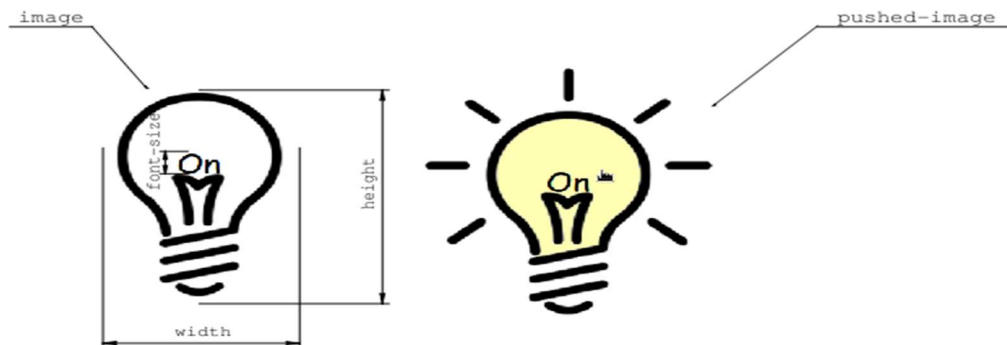


Рисунок 3.47 - Графическая кнопка

Обеспечивает отрисовку графической кнопки (Рис. 3.47) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.66 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина кнопки
height	Число	0	Высота кнопки
image	Строка	undefined	Изображение
pushed-image	Строка	undefined	Изображение при клике
name	Строка	undefined	Имя тега

Таблица 3.67 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="button-image" x=100 y=100 width=200 height=200 pushed-image="true.png" image="false.png" name="tag1"></div></code>
JavaScript	<code>PL.GUI.createButtonImage({x: 100, y: 100, width: 200, height: 200, 'pushed-image': 'true.png' , name: "tag1"});</code>

Таблица 3.68 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
text	Строка	Пустая строка	Текст графической кнопки
font-size	Число	14	Размер текста графической кнопки
font-family	Строка	Arial	Шрифт текста графической кнопки

text-anchor	Строка	middle	Смещение текста по горизонтали
alignment-baseline	Строка	middle	Смещение текста по вертикали
font-stroke	Строка	black	Цвет границы текста графической кнопки
font-stroke-width	Число	0	Ширина границы текста графической кнопки
font-color	Строка	black	Цвет текста графической кнопки
preserve-aspect-ratio	Строка	none	Масштабирование изображения Возможные варианты: xMinYMin xMidYMin xMaxYMin xMinYMid xMidYMid xMaxYMid xMinYMax xMidYMax xMaxYMax
stroke-width	Число	0	Толщина границы изображения
stroke	Строка	black	Цвет границы изображения
timeout	Число	1000	Период обновления (мс)

Таблица 3.69 – Preserve Aspect Ratio

Параметр	Описание
xMin	Выравнивание в соответствии с минимальным значением x viewBox – левая граница области просмотра
xMid	Выравнивание в соответствии с центральной точкой по x viewBox – центр по оси X в области просмотра
xMax	Выравнивание в соответствии с максимальным значением x viewBox – правая граница области просмотра
YMin	выровнять в соответствии с минимальным значением y viewBox – верхняя граница области просмотра
YMid	выровнять в соответствии с центральной точкой по y viewBox – центр по оси Y в области просмотра
YMax	выровнять в соответствии с максимальным значением y viewBox – нижняя граница области просмотра

Таблица 3.70 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="button-image" x=100 y=100 width=200 height=200 pushed-image="true.png" image="false.png" name="tag1" text="JET LOGIC" font-size="40" font-family="Comic Sans MS" text-anchor="start" alignment-baseline="start" font-stroke="#FFFFFF" font-stroke-width="1" font-color="#FF0000" preserve-aspect-ratio="xMaxYMax"></div></pre>
JavaScript	<pre>PL.GUI.createButtonImage({x: 100, y: 100, width: 200, height: 200, 'pushed-image': 'true.png', name: 'tag1', text: 'JET LOGIC', 'font-size': '40', 'font-</pre>



```
family': 'Comic Sans MS', 'text-anchor':  
'end', 'alignment-baseline': 'end', 'font-stroke':  
'#FFFFFF', 'font-stroke-width': '1', 'font-stroke-  
color': '#FF0000', 'font-stroke-color': '#FF0000'}));
```



3.18 Круговая диаграмма

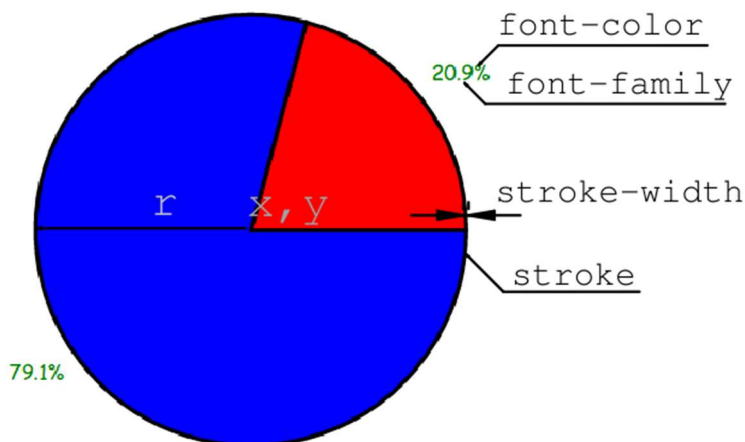


Рисунок 3.48 - Круговая диаграмма

Обеспечивает отрисовку круговой диаграммы (Рис. 3.48) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.71 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса центра круговой диаграммы
y	Число	0	Ордината центра круговой диаграммы
r	Число	0	Радиус диаграммы
data	Массив	[]	Данные

Таблица 3.72 – Таблица параметров - data

Параметр	Тип значения	Значение по умолчанию	Описание
name	Строка	undefined	Имя тега
legend	Строка	undefined	Легенда
fill	Строка	transparent	Цвет сектора

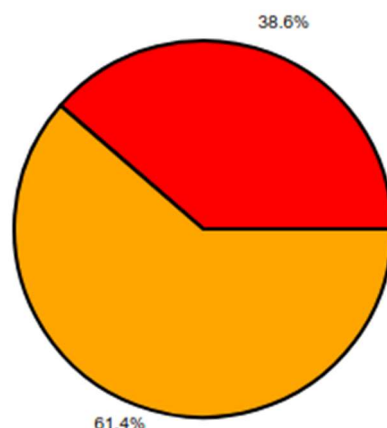


Рисунок 3.49 - Круговая диаграмма с обязательными параметрами

Таблица 3.73 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="pie" x="200" y="450" r="100" data='[{"name": "tag1", "legend": "AI1", "fill": "red"}, {"name": "tag2", "legend": "AI2", "fill": "blue"}]`></div></code>
JavaScript	<code>PL.GUI.createPie({x: 200, y: 450, r: 100, data: '[{"name": "tag1", "legend": "AI1", "fill": "red"}, {"name": "tag2", "legend": "AI2", "fill": "blue"}]'});</code>

Таблица 3.74 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет круговой диаграммы
stroke	Строка	black	Цвет границы круговой диаграммы
stroke-width	Число	1	Ширина границы круговой диаграммы
font-color	Строка	black	Цвет шрифта
font-family	Строка	Arial	Шрифт текста
timeout	Число	1000	Период обновления (мс)
error-fill	Строка	rgba(0,0,0,0.5)	Цвет круговой диаграммы при ошибке

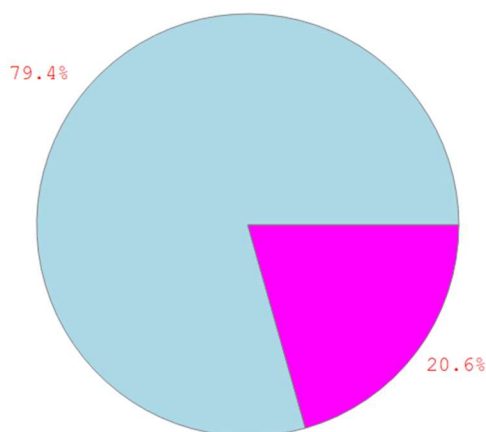


Рисунок 3.50 - Круговая диаграмма с дополнительными параметрами

Таблица 3.75 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="pie" x="200" y="450" r="100" data='[{"name": "tag1", "legend": "AI1", "fill": "red"}, {"name": "tag2", "legend": "AI2", "fill": "blue"}]' stroke="#cccccc" stroke-width="2" font- color="#00FFFF" font-family="Comic Sans MS" timeout="200"></div></pre>
JavaScript	<pre>PL.GUI.createPie({x: 200, y: 450, r: 100, data: '[{"name": "tag1", "legend": "AI1", "fill": "red"}, {"name": "tag2", "legend": "AI2", "fill": "blue"}]', stroke: '#cccccc', 'stroke-width': 2, font- color': '#00FFFF', 'font-family': 'Comic Sans MS', timeout: 200});</pre>



3.19 Кольцевая диаграмма

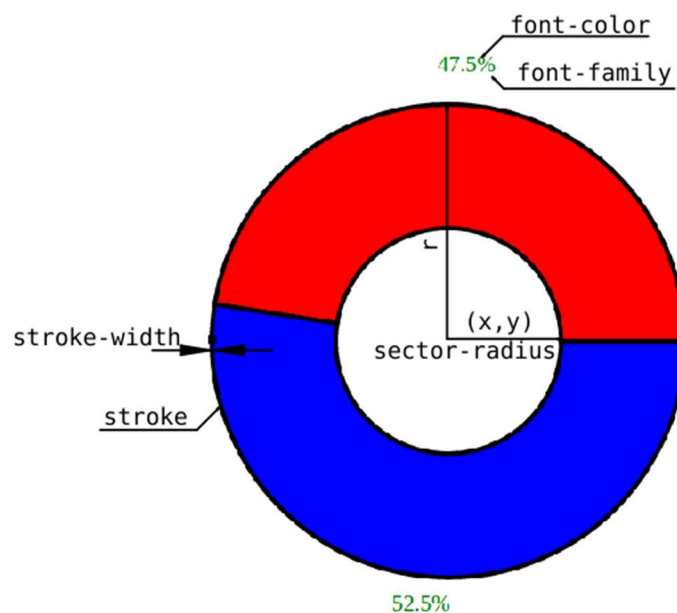


Рисунок 3.51 - Кольцевая диаграмма

Обеспечивает отрисовку кольцевой диаграммы (Рис. 3.51) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.76 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса центра кольцевой диаграммы
y	Число	0	Ордината центра кольцевой диаграммы
r	Число	0	Радиус диаграммы
data	Массив	[]	Данные

Таблица 3.77 – Таблица параметров - data

Параметр	Тип значения	Значение по умолчанию	Описание
name	Строка	undefined	Имя тега
legend	Строка	undefined	Легенда
fill	Строка	transparent	Цвет сектора

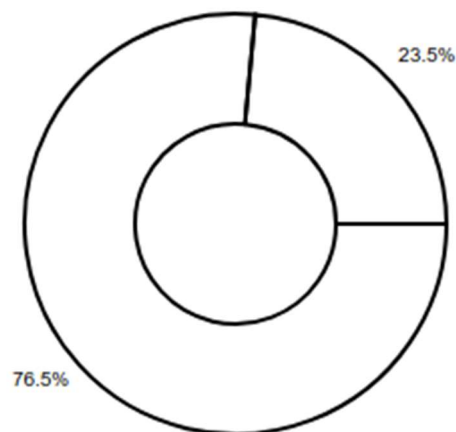


Рисунок 3.52 - Кольцевая диаграмма с обязательными параметрами

Таблица 3.78 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<pre><div class="donut" x="200" y="450" r="100" data='[{"name": "tag1", "legend": "AI1", "fill": "red"}, {"name": "tag2", "legend": "AI2", "fill": "blue"}]' ></div></pre>
JavaScript	<pre>PL.GUI.createDonut({x: 200, y: 450, r: 100, data: '[{"name": "tag1", "legend": "AI1", "fill": "red"}, {"name": "tag2", "legend": "AI2", "fill": "blue"}]'});</pre>

Таблица 3.79 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет кольцевая диаграммы
stroke	Строка	black	Цвет границы кольцевой диаграммы
stroke-width	Число	1	Ширина границы кольцевой диаграммы
font-color	Строка	black	Цвет шрифта
font-family	Строка	Arial	Шрифт текста
sector-radius	Число	0.5	Внутренний радиус круговой диаграммы (от 0 до 1)
timeout	Число	1000	Период обновления (мс)
error-fill	Строка	rgba(0,0,0,0.5)	Цвет кольцевой диаграммы при ошибке

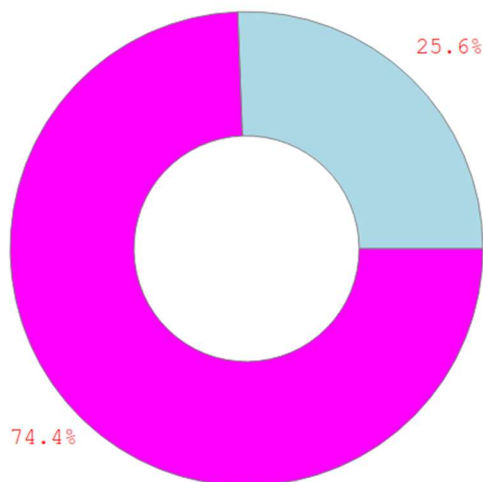


Рисунок 3.53 - Кольцевая диаграмма с дополнительными параметрами

Таблица 3.80 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="donut" x="200" y="450" r="100" data='[{"name": "tag1", "legend":"AI1", "fill": "red"}, {"name": "tag2", "legend":"AI2", "fill": "blue"}]' stroke="#cccccc" stroke-width="2" font-color="#00FFFF" font-family="Comic Sans MS" sector-radius="0.9" timeout="1000"></div></pre>
JavaScript	<pre>PL.GUI.createDonut({x: 200, y: 450, r: 100, data: '[{"name": "tag1", "legend":"AI1", "fill": "red"}, {"name": "tag2", "legend":"AI2", "fill": "blue"}]', stroke: '#cccccc', 'stroke-width': 2, 'font-color': '#00FFFF', 'font-family': 'Comic Sans MS', 'sector-radius': 0.9, timeout: 1000});</pre>



3.20 Столбчатая диаграмма

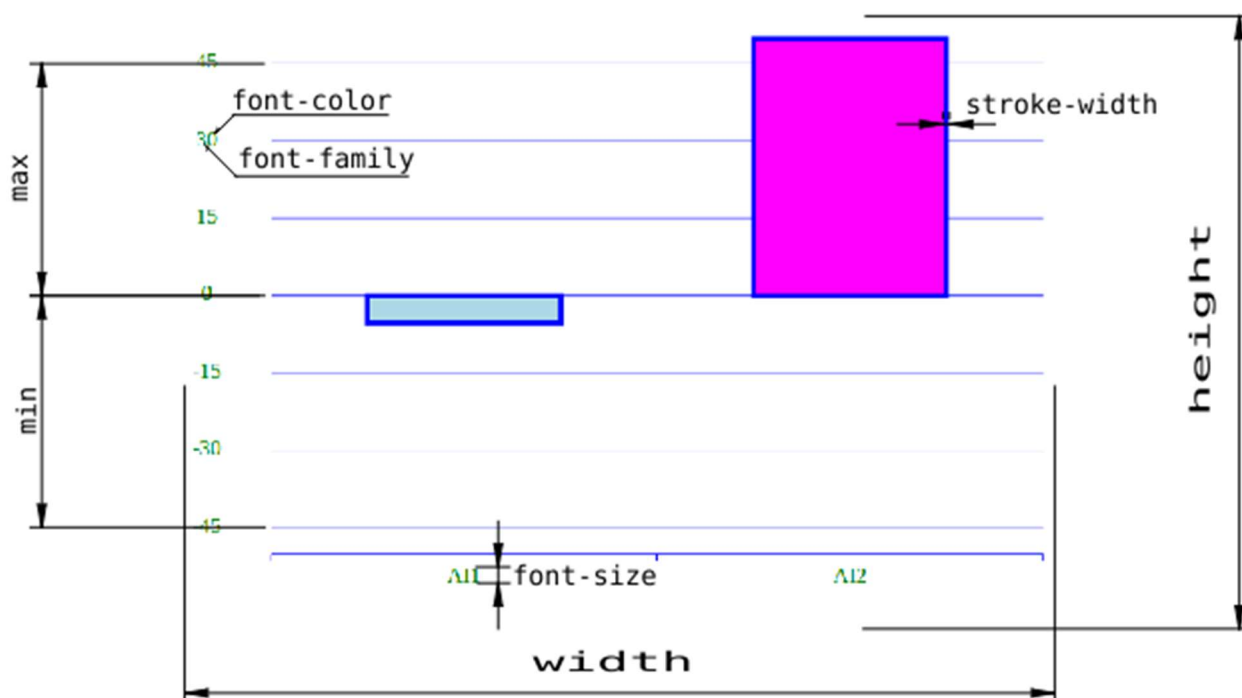


Рисунок 3.54 - Столбчатая диаграмма

Обеспечивает отрисовку столбчатой диаграммы (Рис. 3.54) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.81 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса центра столбчатой диаграммы
y	Число	0	Ордината центра столбчатой диаграммы
data	Массив	[]	Данные
width	Число	0	Ширина столбчатой диаграммы
height	Число	0	Высота столбчатой диаграммы

Таблица 3.82 – Таблица параметров - data

Параметр	Тип значения	Значение по умолчанию	Описание
name	Строка	undefined	Имя тега
legend	Строка	undefined	Легенда
fill	Строка	transparent	Цвет столбца

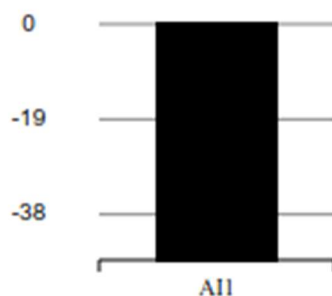


Рисунок 3.55 - Столбчатая диаграмма с обязательными параметрами

Таблица 3.83 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="bars" x="200" y="400" width="300" height="200" data='[{"name": "tag1", "legend": "AI1", "fill": "lightblue"}, {"name": "tag2", "legend": "AI2", "fill": "magenta"}]`></div></code>
JavaScript	<code>PL.GUI.createBars({x: 200, y: 400, width: 300, height: 200, data: '[{"name": "tag1", "legend": "AI1", "fill": "lightblue"}, {"name": "tag2", "legend": "AI2", "fill": "magenta"}]'});</code>

Таблица 3.84 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет столбчатой диаграммы
stroke	Строка	black	Цвет границы столбчатой диаграммы
stroke-width	Число	1	Ширина столбчатой диаграммы
font-color	Строка	black	Цвет шрифта столбчатой диаграммы
font-family	Строка	Arial	Шрифт текста столбчатой диаграммы
font-size	Число	10	Размер текста столбчатой диаграммы
min	Число	undefined	Минимальное значение столбчатой диаграммы
max	Число	undefined	Максимальное значение столбчатой диаграммы
timeout	Число	1000	Период обновления (мс)
error-fill	Строка	rgba(0,0,0,0.5)	Цвет столбчатой диаграммы при ошибке

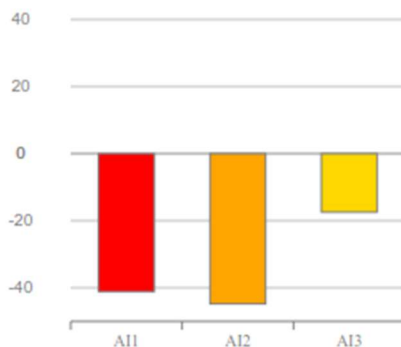


Рисунок 3.56 - Столбчатая диаграмма с обязательными параметрами

Таблица 3.85 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="bars" x="200" y="400" width="300" height="200" data='[{"name": "tag1", "legend": "AI1", "fill": "lightblue"}, {"name": "tag2", "legend": "AI2", "fill": "magenta"}]' stroke="#00FF00" stroke-width="2" font-color="#FF0000" 'font-family': 'Comic Sans MS' font-size="2" min="25" max="20" timeout="200"></div></pre>
JavaScript	<pre>PL.GUI.createBars({x: 200, y: 400, width: 300, height: 200, data: '[{"name": "tag1", "legend": "AI1", "fill": "lightblue"}, {"name": "tag2", "legend": "AI2", "fill": "magenta"}]', stroke: '#00FF00', 'stroke-width': 2, 'font-color': '#FF0000', 'font-family': 'Comic Sans MS', 'font-size': 10, min: 25, max: 20, timeout: 200});</pre>

3.21 Блок вывода значений тега



Рисунок 3.57 - Блок вывода значений тега

Обеспечивает отрисовку блока вывода значений тега (Рис. 3.57) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.86 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса точки привязки текста
y	Число	0	Ордината точки привязки текста
name	Строка	undefined	Имя тега
type	Строка	undefined	Тип выводимого значения (integer, float)

72

Рисунок 3.58 - Блок вывода значений тега с обязательными параметрами

Таблица 3.87 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="label" x=100 y=100 name="AI3" type="integer" timeout="1000"></div></code>
JavaScript	<code>PL.GUI.createLabel({x: 100, y: 100, name: 'AI3', type: 'integer', timeout: 1000});</code>

Таблица 3.88 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет текста
stroke	Строка	black	Цвет границы текста



stroke-width	Число	1	Ширина границы текста
font-family	Строка	Arial	Шрифт текста блока вывода значений тега.
font-size	Число	10	Размер текста блока вывода значений тега
data	Строка	{ }	Отображение текста в зависимости от значений тега
default	Строка	-/-	Выводимое значение, в случае когда значение тега не совпадает ни с одним заданным интервалом
decimal-places	Число	2	Число знаков после запятой для type=«float»
text-anchor	Строка	middle	Смещение текста по горизонтали
alignment-baseline	Строка	middle	Смещение текста по вертикали
timeout	Число	1000	Период обновления (мс)

33

Рисунок 3.59 - Блок вывода значений тега с дополнительными параметрами

Таблица 3.89 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<code><div class="label" x=100 y=100 name="AI3" type="integer" timeout="1000" fill="#cccccc" stroke="#000000" stroke-width="2" font-family="Comic Sans MS" font-size="15" data='{ "LOW": [0,50], "HIGH": [51, 100] }' default='string'></div></code>
JavaScript	<code>PL.GUI.createLabel({x: 100, y: 100, name: 'AI3', type: 'integer', timeout: 1000, stroke: '#000000', 'stroke-width': 2, 'font-family': 'Comic Sans MS', 'font-size': 15, data: '{ "LOW": [0,50], "HIGH": [51, 100] }', 'default': 'string'});</code>

3.22 Выпадающий список

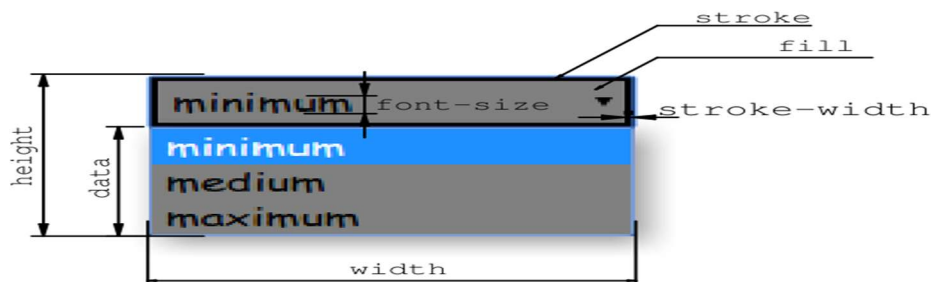


Рисунок 3.60 - Выпадающий список

Обеспечивает отрисовку выпадающего списка (Рис. 3.60) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.90 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина выпадающего списка
height	Число	0	Высота выпадающего списка
name	Строка	undefined	Имя тега
type	Строка	undefined	Тип выводимого значения (integer, float)

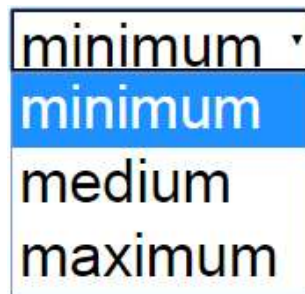


Рисунок 3.61 - Выпадающий список с обязательными параметрами

Таблица 3.91 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="drop-down-list" x=100 y=100 width=100 height=20 data='{ "minimum": 10, "medium": 50, "maximum": 100 }' name="AI3" type="integer"></div></code>
JavaScript	<code>PL.GUI.createDropDownList({x: 100, y: 100, width: 100, height: 20, data: '{ "minimum": 10, "medium": 50, "maximum": 100 }', name: 'AI3', type: 'integer'});</code>

Таблица 3.92 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
----------	--------------	-----------------------	----------



fill	Строка	transparent	Цвет выпадающего списка
stroke	Строка	black	Цвет границы выпадающего списка
stroke-width	Число	0	Ширина выпадающего списка
font-color	Строка	black	Цвет текста выпадающего списка
font-family	Строка	Arial	Шрифт текста выпадающего списка
font-size	Число	14	Размер текста выпадающего списка
timeout	Число	1000	Период обновления (мс)

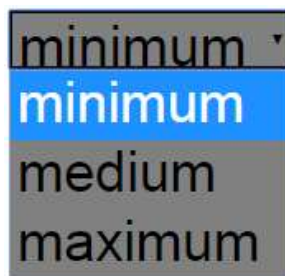


Рисунок 3.62 - Выпадающий список с дополнительными параметрами

Таблица 3.93 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="drop-down-list" x=100 y=100 width=100 height=20 data='{ "minimum": 10, "medium": 50, "maximum": 100}' name="AI3" type="integer" fill="#cccccc" stroke="#000000" stroke-width="2" font-family="Comic Sans MS" font-size="15"></div></pre>
JavaScript	<pre>PL.GUI.createDropDownList({x: 100, y: 100, width: 100, height: 20, data: '{"minimum": 10, "medium": 50, "maximum": 100}', name: 'AI3', type: 'integer', fill: '#cccccc', stroke: '#000000', 'stroke-width': 2, 'font-family': 'Comic Sans MS', 'font-size': 15});</pre>

3.23 Поле ввода значений тега

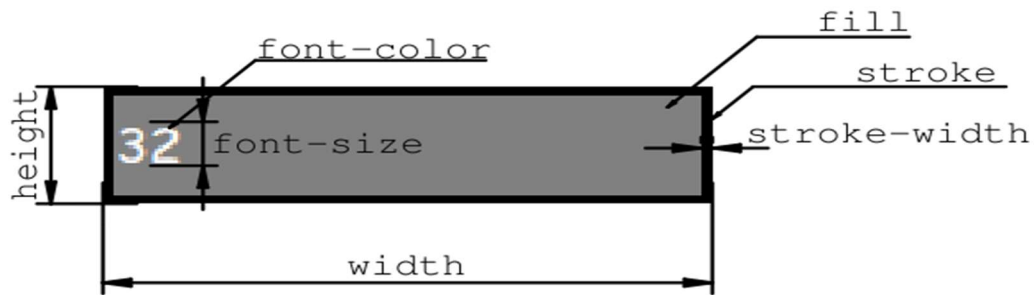


Рисунок 3.63 - Поле ввода значений тега

Обеспечивает отрисовку поле ввода значений тега (Рис. 3.63) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.94 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина поля ввода значений тега
height	Число	0	Высота поля ввода значений тега
name	Строка	undefined	Имя тега
type	Строка	undefined	Тип вводимого значения (integer, float)



Рисунок 3.64 - Поле ввода значений тега с обязательными параметрами

Таблица 3.95 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="edit" x=100 y=100 width=100 height=20 name="AI3" type="integer"></div></code>
JavaScript	<code>PL.GUI.createEdit({x: 100, y: 100, width: 100, height: 20, name: 'AI3', type: 'integer'});</code>

Таблица 3.96 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет поля ввода значений тега
stroke	Строка	black	Цвет границы поля ввода значений тега
stroke-width	Число	0	Ширина поля ввода значений тега



font-color	Строка	black	Цвет текста поля ввода значений тега
font-family	Строка	Arial	Шрифт текста поля ввода значений тега
decimal-places	Число	2	Число знаков после запятой для type=«float»
error-fill	Строка	rgba(0,0,0,0.5)	Цвет поля ввода значений тега при ошибке
timeout	Число	1000	Период обновления (мс)

53.064

Рисунок 3.65 - Поле ввода значений тега с дополнительными параметрами

Таблица 3.97 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="edit" x=100 y=100 width=100 height=20 name="AI3" type="integer" fill: '#cccccc' stroke="#000000" stroke-width="2" font-family="Comic Sans MS"></div></pre>
JavaScript	<pre>PL.GUI.createEdit({x: 100, y: 100, width: 100, height: 20, name: 'AI3', type: 'integer', fill: '#cccccc', stroke: '#000000', 'stroke-width': 2, 'font-family': 'Comic Sans MS'});</pre>



3.24 Термометр

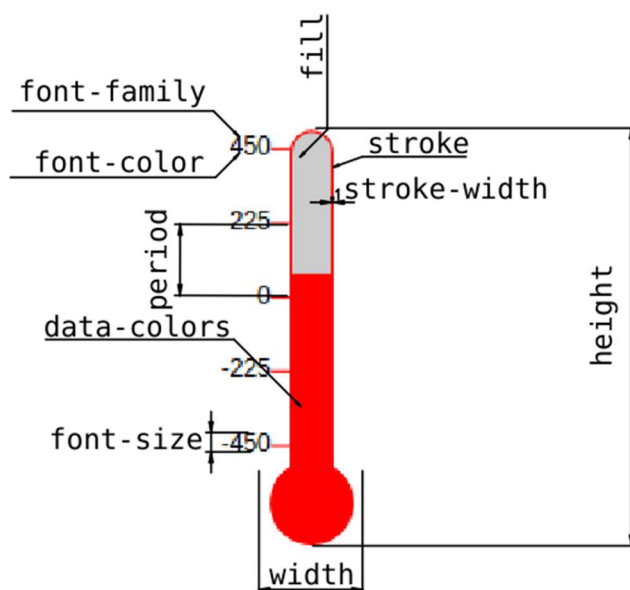


Рисунок 3.66 - Термометр

Обеспечивает отрисовку термометра (Рис. 3.66) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.98 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина термометра
height	Число	0	Высота термометра
name	Строка	undefined	Имя тега
type	Строка	undefined	Тип выводимого значения (integer, float)

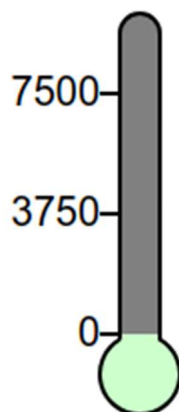


Рисунок 3.67 – Термометр с обязательными параметрами

Таблица 3.99 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="thermometer" x=100 y=100 width=20 height=100 name="AI3" type="integer"></div></code>
JavaScript	<code>PL.GUI.createThermometer({x: 100, y: 100, width: 20, height: 100, name: 'tag1', type: 'integer'});</code>

Таблица 3.100 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет термометра
stroke	Строка	black	Цвет границы термометра
stroke-width	Число	0	Ширина границы термометра
font-color	Строка	black	Цвет текста термометра
font-family	Строка	Arial	Шрифт текста термометра
font-size	Число	14	Размер текста термометра
period	Число	undefined	Период значений термометра
data-colors	Массив	<code>['rgba(0,255,0,0.5)', 'rgba(255,255,0,0.5)', 'rgba(255,0,0,0.5)']</code>	Цвет шкалы термометра в зависимости от полученных значений
min-value	Число	0	Минимальное значение тега
max-value	Число	10000	Максимальное значение тега
error-fill	Строка	<code>rgba(0,0,0,0.5)</code>	Цвет термометра при ошибке
timeout	Число	1000	Период обновления (мс)

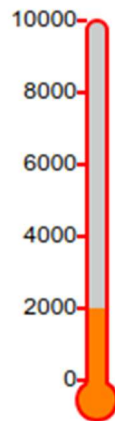


Рисунок 3.68 – Термометр с дополнительными параметрами

Таблица 3.101 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="thermometer" x=100 y=100 width=20 height=100 name="AI3" type="integer" fill="#cccccc" stroke="#000000" stroke-width="2" font-family="Comic Sans MS" font-size="15" period="2" data- colors="{&quot;rgba(255,0,0)&quot;;:[0,1500], &quot;rgba(255,128,0)&quot;;:[1501,3000]}" min- value="500" max-value="5000"></div></pre>
JavaScript	<pre>PL.GUI.createThermometer({x: 100, y: 100, width: 20, height: 100, name: 'tag1', type: 'integer', fill: '#cccccc', stroke: '#000000', 'stroke-width': 2, 'font-family': 'Comic Sans MS', font-size="15", period: 2, 'data-colors': '{&quot;rgba(255,0,0)&quot;;:[0,1500], &quot;rgba(255,128,0)&quot;;:[1501,3000]}' , 'min- value': 500, 'max-value': 5000});</pre>

3.25 Спидометр

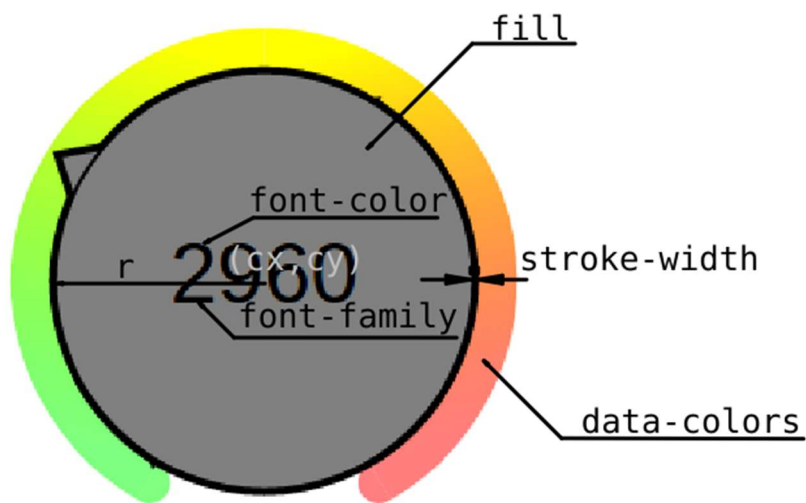


Рисунок 3.69 - Спидометр

Обеспечивает отрисовку спидометра (Рис. 3.69) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.102 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
cx	Число	0	Абсцисса центра спидометра
cy	Число	0	Ордината центра спидометра
r	Число	0	Радиус спидометра
name	Строка	undefined	Имя тега
type	Строка	undefined	Тип выводимого значения (integer, float)

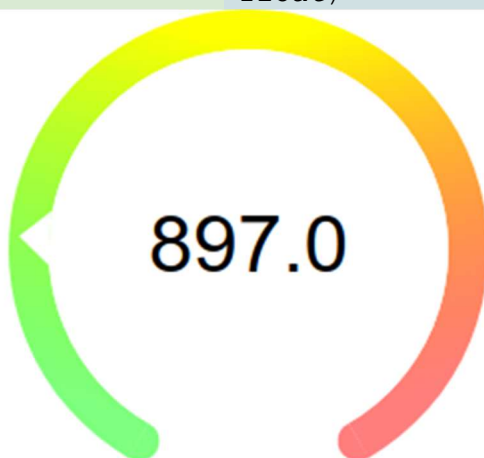


Рисунок 3.70 – Спидометр с обязательными параметрами



Таблица 3.103 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="meter" cx="50" cy="70" r="30" name="tag1" type="integer"></div></code>
JavaScript	<code>PL.GUI.createMeter({cx: 50, cy: 70, r: 30, name: 'tag1', type: 'integer'});</code>

Таблица 3.104 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет спидометра
stroke	Строка	black	Цвет границы спидометра
stroke-width	Число	0	Ширина границы спидометра
font-color	Строка	black	Цвет текста спидометра
font-family	Строка	Arial	Шрифт текста спидометра
data-colors	Массив	<code>['rgba(0,255,0,0.5)', 'rgba(255,255,0,0.5)', 'rgba(255,0,0,0.5)']</code>	Цвет шкалы спидометра в зависимости от полученных значений
min-value	Число	0	Минимальное значение тега
max-value	Число	10000	Максимальное значение тега
error-fill	Строка	<code>rgba(0,0,0,0.5)</code>	Цвет спидометра при ошибке
timeout	Число	1000	Период обновления (мс)

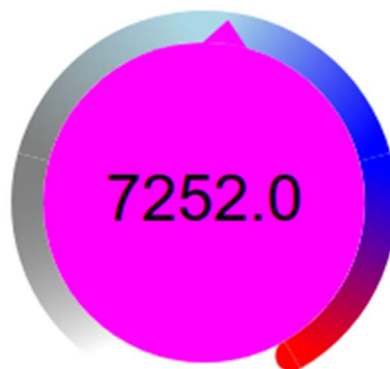


Рисунок 3.71 – Спидометр с дополнительными параметрами

Таблица 3.105 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="meter" cx="50" cy="70" r="30" name="tag1" type="integer" fill="#cccccc" stroke="#000000"></code>



JavaScript

```
stroke-width="2" font-family="Comic Sans MS" data-  
colors="{&quot;rgba(255,0,0)&quot;:[0,1500],  
&quot;rgba(255,128,0)&quot;:[1501,3000]}" min-  
value="500" max-value="5000"></div>  
PL.GUI.createMeter({cx: 50, cy: 70, r: 30, name:  
'tag1', type: 'integer', fill: '#cccccc', stroke:  
'#000000', 'stroke-width': 2, 'font-family': 'Comic  
Sans MS', 'data-colors':  
'{&quot;rgba(255,0,0)&quot;:[0,1500],  
&quot;rgba(255,128,0)&quot;:[1501,3000]}' , min-  
value="500" max-value="5000"});
```

3.26 Графический индикатор значений тега

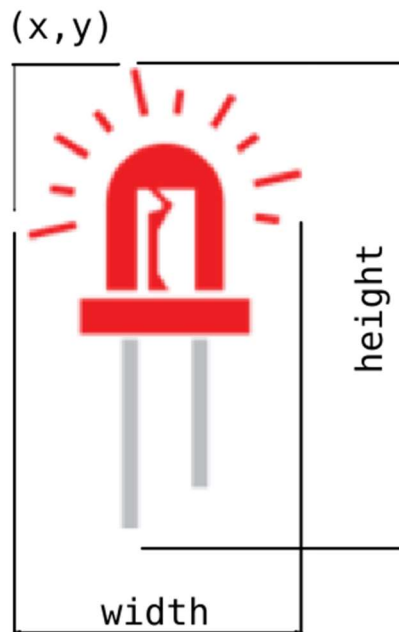


Рисунок 3.72 - Графический индикатор значений тега

Обеспечивает отрисовку графического индикатора значений тега (Рис. 3.72) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.106 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина графического индикатора значений тега
height	Число	0	Высота графического индикатора значений тега
name	Строка	undefined	Имя тега
default	Строка	undefined	Изображение по умолчанию
data	Строка	{ }	Данные

Таблица 3.107 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="value-image" x=100 y=30 width=60 height=60 data='{ "1.jpg": 1, "2.jpg": 2 }' name="AI3" default="3.png"></div></code>
JavaScript	<code>PL.GUI.createValueImage({x: 100, y: 30, width: 60, height: 60, data: '{ "1.jpg": 1, "2.jpg": 2 }', name: 'AI3', default: '3.png'});</code>

Таблица 3.108 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
preserve-aspect-ratio	Строка	none	Масштабирование изображения Возможные варианты: xMinYMin xMidYMin xMaxYMin xMinYMid xMidYMid xMaxYMid xMinYMax xMidYMax xMaxYMax
error-fill	Строка	rgba(0,0,0,0.5)	Цвет графического индикатора при ошибке
stroke-width	Число	0	Толщина границы изображения
stroke	Строка	black	Цвет границы изображения
timeout	Число	1000	Период обновления (мс)

Таблица 3.109 – Preserve Aspect Ratio

Параметр	Описание
xMin	Выравнивание в соответствии с минимальным значением x viewBox – левая граница области просмотра
xMid	Выравнивание в соответствии с центральной точкой по x viewBox – центр по оси X в области просмотра
xMax	Выравнивание в соответствии с максимальным значением x viewBox – правая граница области просмотра
YMin	выровнять в соответствии с минимальным значением y viewBox – верхняя граница области просмотра
YMid	выровнять в соответствии с центральной точкой по y viewBox – центр по оси Y в области просмотра
YMax	выровнять в соответствии с максимальным значением y viewBox – нижняя граница области просмотра

Таблица 3.110 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="value-image" x=100 y=30 width=60 height=60 data='{ "1.jpg": 1, "2.jpg": 2 }' name="AI3" default="3.png" preserve-aspect-ratio="xMidYMid" stroke-width="2" stroke="#000000" error-fill="rgba(0,0,0,1)"></div></pre>
JavaScript	<pre>PL.GUI.createValueImage({x: 100, y: 30, width: 60, height: 60, data: '{ "1.jpg": 1, "2.jpg": 2 }', name: 'AI3', default: '3.png', 'preserve-aspect-ratio': 'xMaxYMin', 'stroke-width': 2, stroke: '#000000', 'error-fill': ' rgba(0,0,0,1) '});</pre>

3.27 Графический индикатор интервальных значений тега

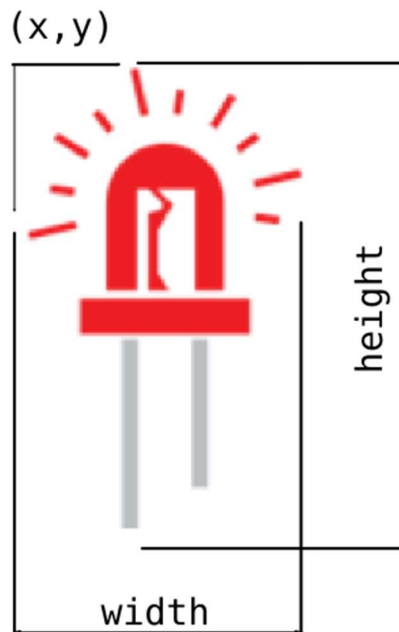


Рисунок 3.73 - Графический индикатор интервальных значений тега

Обеспечивает отрисовку графического индикатора интервальных значений тега (Рис. 3.73) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.111 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина графического индикатора интервальных значений тега
height	Число	0	Высота графического индикатора интервальных значений тега
name	Строка	undefined	Имя тега
default	Строка	undefined	Изображение по умолчанию
data	Строка	{ }	Данные

Таблица 3.112 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<pre><div class="interval-image" x=100 y=30 width=60 height=60 data='{ "1.jpg": 1, "2.jpg": 2 }' name="AI3" default="3.png"></div></pre>
JavaScript	<pre>PL.GUI.createIntevalImage({x: 100, y: 30, width: 60, height: 60, data: '{ "1.jpg": 1, "2.jpg": 2 }', name: 'AI3', default: '3.png'});</pre>

Таблица 3.113 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
preserve-aspect-ratio	Строка	none	Масштабирование изображения Возможные варианты: xMinYMin xMidYMin xMaxYMin xMinYMid xMidYMid xMaxYMid xMinYMax xMidYMax xMaxYMax
error-fill	Строка	rgba(0,0,0,0.5)	Цвет графического индикатора при ошибке
stroke-width	Число	0	Толщина границы изображения
stroke	Строка	black	Цвет границы изображения
timeout	Число	1000	Период обновления (мс)

Таблица 3.114 – Preserve Aspect Ratio

Параметр	Описание
xMin	Выравнивание в соответствии с минимальным значением x viewBox – левая граница области просмотра
xMid	Выравнивание в соответствии с центральной точкой по x viewBox – центр по оси X в области просмотра
xMax	Выравнивание в соответствии с максимальным значением x viewBox – правая граница области просмотра
YMin	выровнять в соответствии с минимальным значением y viewBox – верхняя граница области просмотра
YMid	выровнять в соответствии с центральной точкой по y viewBox – центр по оси Y в области просмотра
YMax	выровнять в соответствии с максимальным значением y viewBox – нижняя граница области просмотра

Таблица 3.115 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<pre><div class="interval-image" x=100 y=30 width=60 height=60 data='{ "1.jpg": 1, "2.jpg": 2 }' name="AI3" default="3.png" preserve-aspect-ratio="xMidYMid" stroke-width="2" stroke="#000000" error-fill="rgba(0,0,0,1)"></div></pre>
JavaScript	<pre>PL.GUI.createIntervalImage({x: 100, y: 30, width: 60, height: 60, data: '{ "1.jpg": 1, "2.jpg": 2 }', name: 'AI3', default: '3.png', 'preserve-aspect-ratio': 'xMaxYMin', 'stroke-width': 2, stroke: '#000000', 'error-fill': ' rgba(0,0,0,1) '});</pre>



3.28 Линейный график

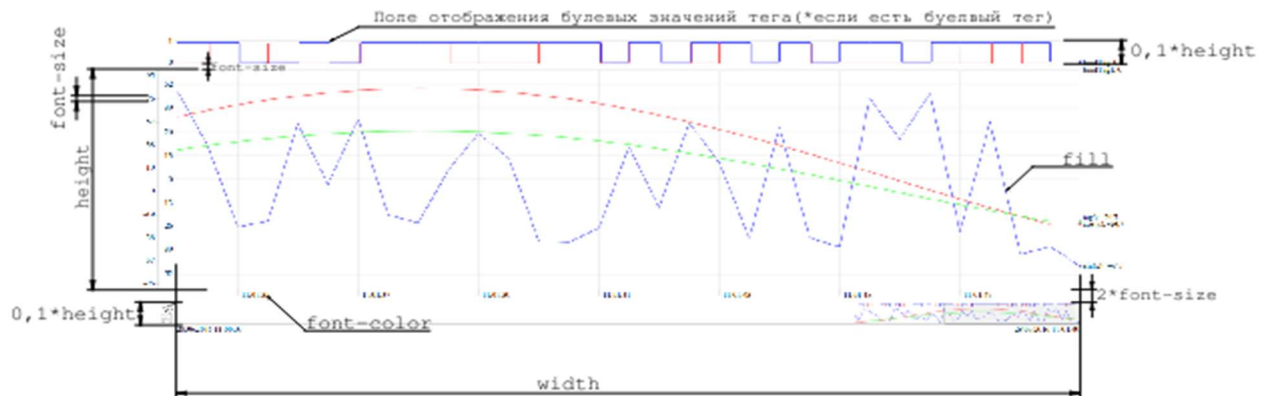


Рисунок 3.74 - Линейный график

Обеспечивает отрисовку линейного графика (Рис. 3.74) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.116 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла графика
y	Число	0	Ордината верхнего левого угла графика
width	Число	0	Ширина графика
height	Число	0	Высота графика
data	Строка	{ }	Данные графиков (Поле позволяет сохранять настройки графика в файл и применением этих настроек)

Таблица 3.117 – Параметры для использования в поле 'data'

Описание
"name" - имя опрашиваемого тега (обязательно к заполнению)
"bitNumder" (опционально) - номер бита



"legend" - имя графика

"fill" - цвет линии графика (по умолчанию stroke или global.window.STROKE)

"min" - минимальное значение тега (для булевых 0)

"max" - максимальное значение тега (для булевых 1)

"visible" - скрыть/отобразить (по умолчанию - отобразить)

"width" - толщина графика (по умолчанию line-width (если не указано, то global.window.STROKE_WIDTH), в относительных единицах(как stroke-width, grid-width))

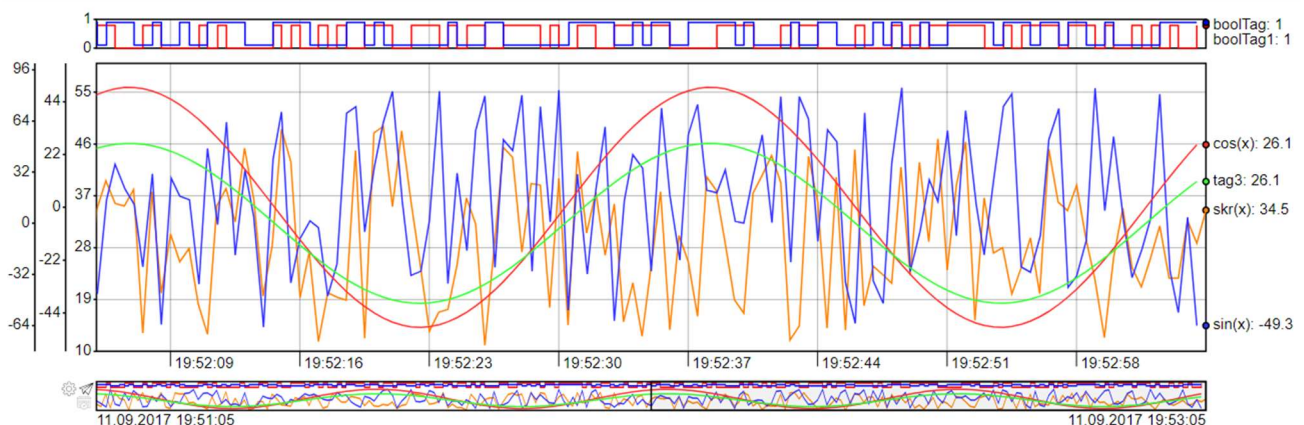


Рисунок 3.75 – Линейный график с обязательными параметрами

Таблица 3.118 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<pre><div class="line-chart" x="100" y="250" width="400" height="200" data='[{"name": "tag2", "legend": "cos(x)", "fill": "#ff3333", "min": -60, "max": 60}, {"name": "tag3", "fill": "#33ff33", "min": -80, "max": 100}]' timeout="1000"></div></pre>
JavaScript	<pre>PL.GUI.createLineChart({x: 100, y: 250, width: 400, height: 200 data: '[{"name": "tag2", "legend": "cos(x)", "fill": "#ff3333", "min": -60, "max": 60}, {"name": "tag3", "fill": "#33ff33", "min": -80, "max": 100}]', timeout: "1000" });</pre>

Таблица 3.119 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
fill	Строка	transparent	Цвет заливки компонента
stroke	Строка	black	Цвет границы графика
stroke-width	Число	0	Ширина границы графика
font-color	Строка	black	Цвет текста графика
font-family	Строка	Arial	Шрифт текста графика



grid-width	Число	0.2	Ширина сетки
years-count	Число	1	Максимальная дата архива в годах
line-width	Число	0	Ширина линии графика
time	Число	60000	Размер окна просмотра (time < max-time)
max-time	Число	120000	Максимальное время хранения значения графика
timeout	Число	1000	Период обновления (мс)
error-fill	Строка	rgba(0,0,0,0.5)	Цвет графика при ошибке

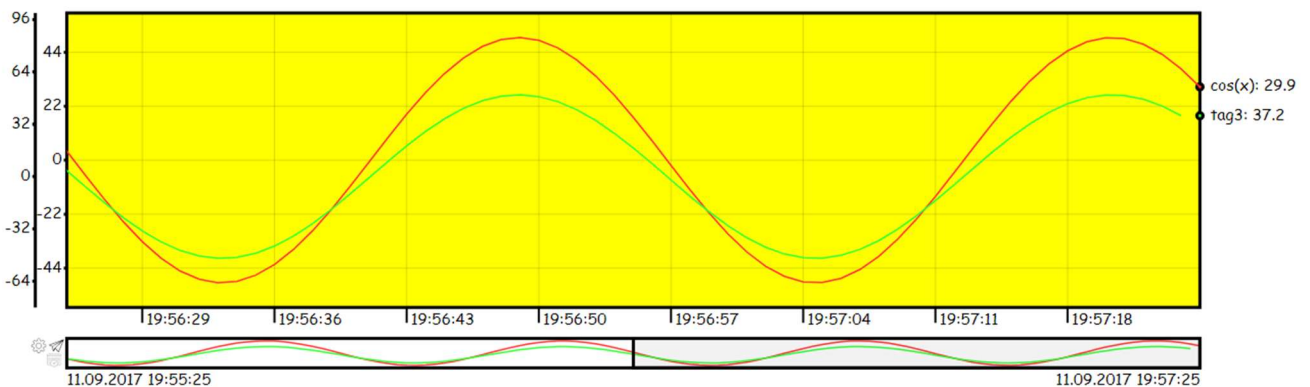


Рисунок 3.76 – Линейный график с дополнительными параметрами

Таблица 3.120 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="line-chart" x="100" y="250" width="400" height="200" data='[{"name": "tag2", "legend": "cos(x)", "fill": "#ff3333", "min": -60, "max": 60}, {"name": "tag3", "fill": "#33ff33", "min": -80, "max": 100}]' timeout="1000" stroke="#000000" grid-width="0.1" years-count="5" stroke-width="2" font-family="Comic Sans MS" ></div></pre>
JavaScript	<pre>PL.GUI.createLineChart({x: 100, y: 250, width: 400, height: 200 data: '[{"name": "tag2", "legend": "cos(x)", "fill": "#ff3333", "min": -60, "max": 60}, {"name": "tag3", "fill": "#33ff33", "min": -80, "max": 100}]', timeout: "1000", stroke: '#000000', 'grid-width': 0.1, 'years-count': 5, 'stroke-width': 2, 'font-family': 'Comic Sans MS'});</pre>

1.1.1 Просмотр архивных данных значений тега

Существует 2 режима просмотра значений на графике, а именно просмотр в режиме реального времени и просмотр архивных данных. По умолчанию активируется режим реального времени. Для просмотра архивных данных необходимо кликнуть курсором мыши на иконку календаря , затем в появившемся окне (см. рис. 3.77) выбрать дату и далее (см. рис. 3.78) выбрать время. После этого произойдет подгрузка данных из устройства, график будет неподвижным. По клику на иконку бумажного самолета  график перейдет обратно в режим реального времени.

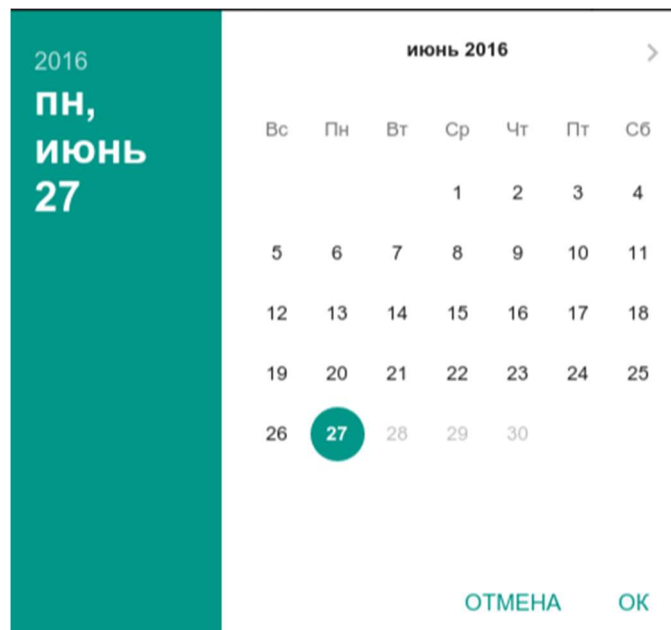


Рисунок 3.77 – Окно выбора даты



Рисунок 3.78 – Окно выбора времени

1.1.2 Настройка графиков

Параметр data хранит данные по каждому конкретному графику. В компоненте предусмотрена возможность настройки этих данных. Необходимо кликнуть курсором мыши на иконку шестеренки . В появившемся окне (см. рис. 3.79) представлен список текущих графиков (серий). По клику на название серии откроется окно редактирования параметров серии (см. рис. 3.80). Изменение цвета и толщины линии происходит динамически (не требуется перерисовывать компонент заново), редактирование минимальных и максимальных значений (также как и удаление серии в целом) повлияет на шкалы графика и потребует их пересчета с последующей полной перерисовкой компонента. Кнопка «Добавить график» позволяет добавить еще одну серию в компонент. Данные, которые требуются для нового графика, аналогичны параметрам в поле 'data'. Предусмотрена возможность сохранения (и загрузки) настроечных данных в формате json в файл lineChartSettings.json. Настройки можно также редактировать напрямую в файле (в соответствии с настройкой поля 'data').



1. можно выбрать отображение графиков, добавить новый график, загрузить данные для графика из файла.

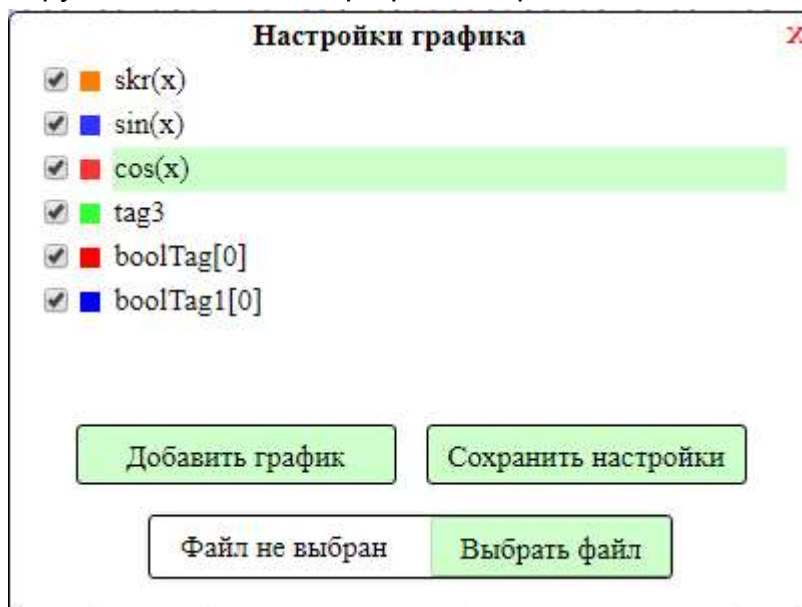


Рисунок 3.79 – Настройка графиков



Рисунок 3.80 – Настройка графика

3.29 Флажок

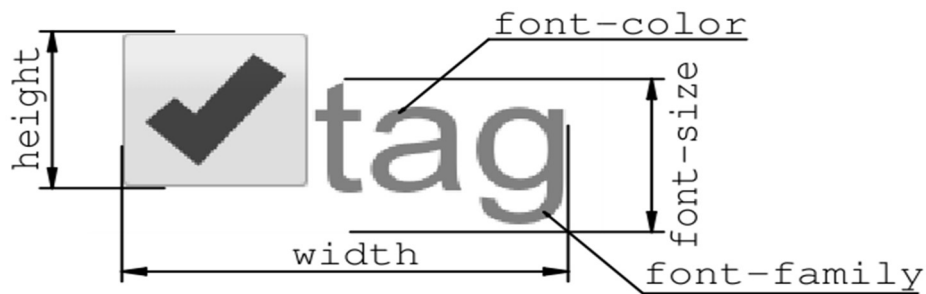


Рисунок 3.81 - Флажок

Обеспечивает отрисовку флажка (Рис. 3.81) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.121 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина флажка
height	Число	0	Высота флажка
name	Строка	undefined	Имя тега
bit-number	Число	0	Номер бита



Рисунок 3.82 – Флажок с обязательными параметрами

Таблица 3.122 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="checkbox" x=50 y=10 height="10" name="AI3" bit-number="1"></div></code>
JavaScript	<code>PL.GUI.createCheckbox({x: 50, y: 10, height: 10, name: 'AI3', 'bit-number': 1});</code>

Таблица 3.123 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
font-color	Строка	black	Цвет текста флажка
font-family	Строка	Arial	Шрифт текста флажка
font-size	Число	14	Размер текста флажка
text	Строка	-	Текст флажка



error-fill	Строка	rgba(0,0,0,0.5)	Цвет флажка при ошибке
timeout	Число	1000	Период обновления (мс)



Рисунок 3.83 – Флажок с дополнительными параметрами

Таблица 3.124 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="checkbox" x=50 y=10 height="10" name="AI3" bit-number="1" font-color="#FF0000" font-family="Comic Sans MS" font-size="10" text="String" error-fill="rgba(255,0,0,0.5)"></div></pre>
JavaScript	<pre>PL.GUI.createCheckbox({x: 50, y: 10, height: 10, name: 'AI3', 'bit-number': 1, 'font-color': '#FF0000', 'font-family': 'Comic Sans MS', 'font-size': 10, text: 'String', 'error-fill': 'rgba(255,0,0,0.5)'});</pre>

3.30 Переключатель

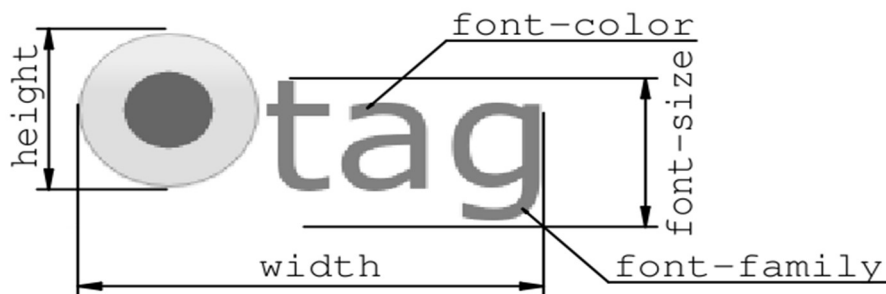


Рисунок 3.84 – Переключатель

Обеспечивает отрисовку переключателя (Рис. 3.84) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.125 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
width	Число	0	Ширина флажка
height	Число	0	Высота флажка
name	Строка	undefined	Имя тега
value	Число	undefined	Значение тега



Рисунок 3.85 – Переключатель с обязательными параметрами

Таблица 3.126 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="radio-button" x=50 y=10 height="10" name="AI3" type="integer" value="30"></div></code>
JavaScript	<code>PL.GUI.createRadioButton({x: 50, y: 10, height: 10, name: 'AI3', type: 'integer', value: 30});</code>

Таблица 3.127 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
font-color	Строка	black	Цвет текста флажка
font-family	Строка	Arial	Шрифт текста флажка
font-size	Число	14	Размер текста флажка
text	Строка	undefined	Текст флажка



error-fill	Строка	rgba(0,0,0,0.5)	Цвет флажка при ошибке
timeout	Число	1000	Период обновления (мс)



Рисунок 3.86 – Переключатель с дополнительными параметрами

Таблица 3.128 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="radio-button" x=50 y=10 height="10" name="AI3" type="integer" value="30" font-color="#FF0000" font-family="Comic Sans MS" font-size="10" text="String" error-fill="rgba(255,0,0,0.5)"></div></pre>
JavaScript	<pre>PL.GUI.createRadioButton({x: 50, y: 10, height: 10, name: 'AI3', type: 'integer', value: 30, 'font-color': '#FF0000', 'font-family': 'Comic Sans MS', 'font-size': 10, text: 'String', 'error-fill': 'rgba(255,0,0,0.5)'});</pre>



3.31 Поле вывода ошибок

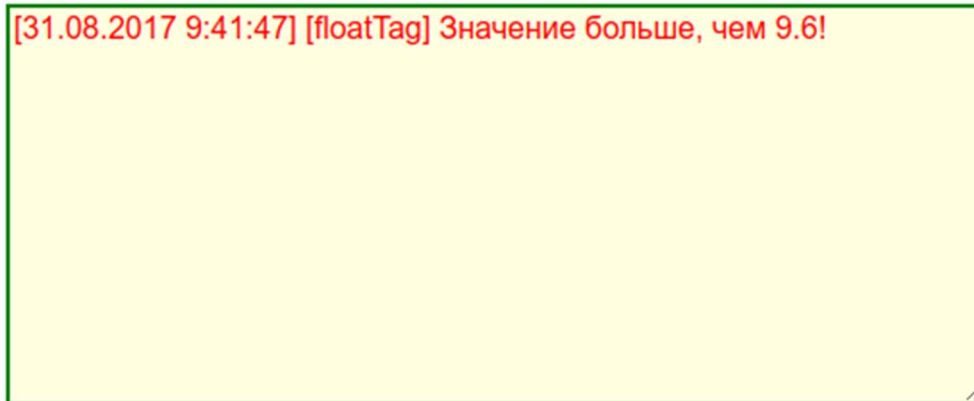


Рисунок 3.87 - Поле вывода ошибок

Обеспечивает поля вывода ошибок (Рис. 3.87) на графическом интерфейсе пользователя в формате SVG.

Таблица 3.129 – Таблица обязательных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
height	Число	0	Высота поля вывода ошибок
width	Число	0	Ширина поля вывода ошибок
data	Массив	[]	Данные

Таблица 3.130 – Параметры для использования в поле 'data'

Описание
"name" – имя опрашиваемого тега (обязательно к заполнению)
"callback" – строка, содержащая функцию обратного вызова. В аргументы передается value (значение опрашиваемого тега). Функция должна возвращать true, если текущее значение считается ошибочным и false в обратном случае
"text" – текст, выводимый в случае возникновения ошибки

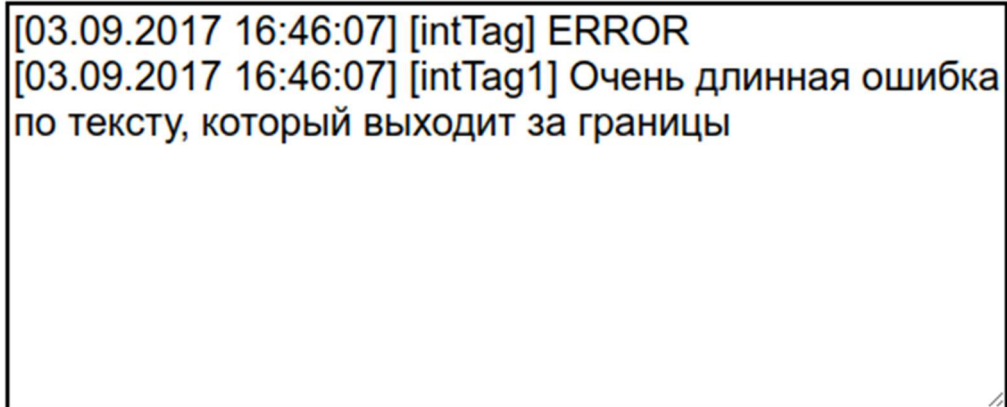


Рисунок 3.88 - Поле вывода ошибок с обязательными параметрами

Таблица 3.131 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="error-area" x="10" y="25" width="200" height="50" data='[{"name": "intTag", "callback": "function(value){return value>0;}", "text": "ERROR"}]'></div></code>
JavaScript	<code>PL.GUI.createErrorArea({x: 10, y: 25, width: 200, height: 50, data: '[{"name": "intTag", "callback": "function(value){return value>0;}", "text": "ERROR"}]' });</code>

Таблица 3.132 – Таблица дополнительных параметров

Параметр	Тип значения	Значение по умолчанию	Описание
font-color	Строка	black	Цвет текста поля вывода ошибок
font-family	Строка	Arial	Шрифт текста поля вывода ошибок
font-size	Число	14	Размер текста поля вывода ошибок
stroke-width	Число	1	Толщина границы поля вывода ошибок
error-fill	Строка	rgba(0,0,0,0.5)	Цвет поля вывода ошибок при ошибке
fill	Строка	white	Цвет поля вывода ошибок
border-color	Строка	black	Цвет границы поля вывода ошибок
stroke	Строка	black	Цвет границы поля вывода ошибок
timeout	Число	1000	Период обновления (мс)



[03.09.2017 16:47:28] [floatTag] Значение больше, чем 9.6!

Рисунок 3.89 - Поле вывода ошибок с дополнительными параметрами

Таблица 3.133 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<pre><div class="error-area" x="10" y="25" width="200" height="50" data='[{"name": "intTag", "callback": "function(value){return value>0;}", "text": "ERROR"}]' font-color="red" family="Comic Sans MS" font-size="10" error-fill="rgba(255,255,0, 0.5)" fill="green" border-color="blue" stroke="magenta"></div></pre>
JavaScript	<pre>PL.GUI.createErrorArea({x: 10, y: 25, width: 200, height: 50, data: '[{"name": "intTag", "callback": "function(value){return value>0;}", "text": "ERROR"}]', , 'font-color': "red", 'font-family': 'Comic Sans MS', 'font-size':10, 'error-fill': "rgba()255,255,0,0.5", 'fill': "green", 'border-color': "blue", stroke: magenta});</pre>



3.32 Журнал событий (log)

Данный компонент использует журнал событий, который ведется модулями PL302/PL307 на аппаратном уровне (более подробно о настройке журнала событий см. «Руководство по эксплуатации» на устройство). Компонент позволяет отображать сообщения из журнала в виде таблицы. В зависимости от типа события строка таблицы окрашивается в определенный цвет (см. далее `event-colors`).

Также есть возможность отображать как все события, так и события произошедшие с момента последнего квитирования (подтверждения оператором о просмотре).

Дата	Время	Статус	Сообщение
2017-07-31	23:47:22.002	Активно	fifth message
2017-07-31	23:47:22.002	Активно	fifth message
2017-07-31	13:43:04.092	Активно	fourth message
2017-07-31	13:43:04.092	Активно	fourth message
2017-07-31	13:34:04.092	Неактивно	third message
2017-07-31	13:34:04.092	Неактивно	third message

Рисунок 3.90 – Журнал событий

Таблица 3.134 – Параметры компонента log

Параметр	Тип значения	Значение по умолчанию	Описание
Обязательные параметры			
x	Число	0	Абсцисса верхнего левого угла
y	Число	0	Ордината верхнего левого угла
height	Число	0	Высота таблицы
width	Число	0	Ширина таблицы
Дополнительные параметры			
font-color	Строка	black	Цвет шрифта
font-family	Строка	Arial	Шрифт текста
font-size	Число	14	Размер текста
stroke-width	Число	1	Толщина границы журнала событий
fill	Строка	white	Цвет журнала событий
stroke	Строка	black	Цвет границы журнала событий
text-align	Строка	center	Расположение текста в ячейке журнала событий

Параметр	Тип значения	Значение по умолчанию	Описание
event-colors	Массив	{'info': 'blue', 'warning': 'yellow', 'error': 'red'}	Цвета полей журнала событий в зависимости от типа событий
error-fill	Строка	rgba(0,0,0,0.5)	Цвет журнала событий при ошибке
timeout	Число	1000	Период обновления(мс)

Таблица 3.135 – Пример использования обязательных параметров

Тип использования	Пример использования
HTML	<code><div class="log" id="myLog" x="200" y="50" height="100" width="400"></div></code>
JavaScript	<code>PL.GUI.createLog({x: 200, y: 50, width: 400, height: 100});</code>

Таблица 3.136 – Пример использования дополнительных параметров

Тип использования	Пример использования
HTML	<code><div class="log" id="myLog" x="200" y="50" height="100" width="400" font-color="red" font-family="Comic Sans MS" font-size="20" stroke-width="2" fill="green" stroke="green" text-align="center" event-colors='{ "warning": "magenta", "info": "white", "error": "red" }'></div></code>
JavaScript	<code>PL.GUI.createLog({x: 200, y: 50, width: 400, height: 100, 'font-color': "red", 'font-family': "Comic Sans MS", 'font-size': "20", 'stroke-width': "2", fill: "green", stroke: "green", 'text-align': "center", 'event-colors': ['["warning": "magenta", "info": "white", "error": "red"]']});</code>

Дата	Время	Статус	Сообщение
2017-07-31	23:47:22.002	Активно	fifth message
2017-07-31	23:47:22.002	Активно	fifth message
2017-07-31	13:43:04.092	Активно	fourth message
2017-07-31	13:43:04.092	Активно	fourth message
2017-07-31	13:34:04.092	Неактивно	third message
2017-07-31	13:34:04.092	Неактивно	third message

Заблокировать загрузку

Рисунок 3.92 – Журнал событий с дополнительными параметрами



Пример использования журнала событий с блокировкой более старых сообщений.

Дата	Время	Статус	Сообщение
2017-07-31	23:47:22.002	Активно	fifth message
2017-07-31	23:47:22.002	Активно	fifth message
2017-07-31	13:43:04.092	Активно	fourth message
2017-07-31	13:43:04.092	Активно	fourth message
2017-07-31	13:34:04.092	Неактивно	third message
2017-07-31	13:34:04.092	Неактивно	third message

Заблокировать подгрузку

Рисунок 3.93 – Журнал событий с событиями на блокировку подгрузки более старых сообщений

Пример использования

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Сохранение значений тега в файл | Пример использования</title>
    <link rel="stylesheet" type="text/css" href="../../build/pl.min.css">
  </head>
  <body>
    <div id="pl-gui" width=800 height=500>
      <div class="log" id="myLog" x="200" y="50" height="100" width="400"></div>
      <div class="button" id="mybutton" x="325" y="170" height="15" width="150"
text="Заблокировать подгрузку"></div>
    </div>
    <script src="../../build/pl.min.js"></script>
    <script>
      var scrollIsAvailable = true;
      PL.GUI.getComponentById("mybutton").bindEvent('click', function(){
        if(scrollIsAvailable){
          PL.GUI.getComponentById("mybutton").setState({"text": "Разрешить
подгрузку", "fill": "lightgreen"});
        }
        else{
          PL.GUI.getComponentById("mybutton").setState({"text": "Заблокировать
подгрузку", "fill": "lightblue"});
        }

        PL.GUI.getComponentById("myLog").blockUploading(scrollIsAvailable);
        scrollIsAvailable = !scrollIsAvailable;
      });
    </script>
  </body>
</html>
```



3.33 Сохранение значений тега в файл

Элемент позволяющий сохранять и выгружать значения тегов из файлов. Для этого необходимо вызвать функции **PL.saveTagValues** и **PL.loadTagValues** соответственно. В примерах ниже они вызываются при возникновении события клика на кнопку, в общем случае ограничения на способы вызова отсутствуют. Сохранение производится в формате csv в файл `pl_tags.csv`. Функция **PL.saveTagValues** выполняет чтение значений тегов из устройства и их сохранение вне зависимости от того, используются они в других компонентах или нет. Функция **PL.loadTagValues** выполняет загрузку тегов и их установку (запись) в устройство.

Таблица 3.137 – Пример использования функций

Тип использования	Пример использования
PL.saveTagValues	<code>PL.saveTagValues([{"name": "tag1", "description": "first"}, {"name": "tag2", "description": "second"}]);</code>
PL.loadTagValues	<code>PL.loadTagValues();</code>

Таблица 3.138 – Параметры для использования в функции 'PL.saveTagValues'

Описание
"name" – имя опрашиваемого тега (обязательно к заполнению)
"description" – строка, добавляемая к значению в сохраняемом файле для удобства его чтения, содержащая пояснения к данному тегу



Рисунок 3.94 – Сохранение значений тегов

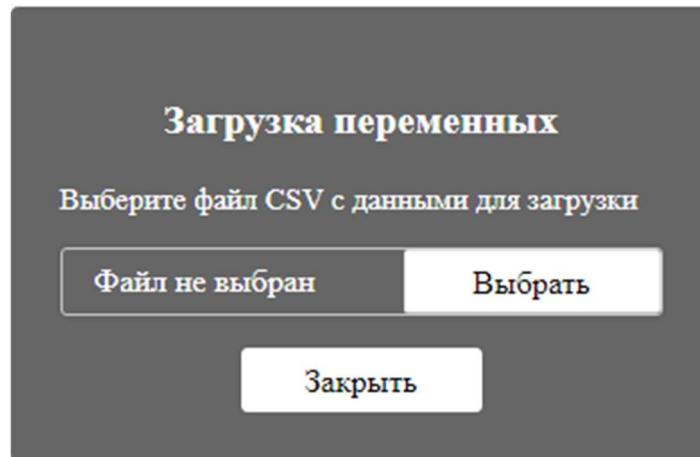


Рисунок 3.95 – Загрузка значений тегов

Таблица 3.139 – Примеры использования сохранения значения тега в файл

Пример использования
<pre> <!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Сохранение значений тега в файл Пример использования</title> <link rel="stylesheet" type="text/css" href="../../build/pl.min.css"> </head> <body> <div id="pl-gui" width=800 height=400 style="height: 920px"> <div id="savebtn" class="button" x="70" y="10" width="60" height="30" text="Save" font-size="10"></div> </div> <script src="../../build/pl.min.js"></script> <script> PL.GUI.getComponentById("savebtn").bindEvent('click', function(){ PL.saveTagValues([{name:"test1", description: "first"}, {name:"test2", description: "second"}]); }); </script> </body> </html> </pre>

Таблица 3.140 – Пример использования выгрузки значения тега из файла

Пример использования
<pre> <!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Сохранение значений тега в файл Пример использования</title> <link rel="stylesheet" type="text/css" href="../../build/pl.min.css"> </head> <body> <div id="pl-gui" width=800 height=400 style="height: 920px"> <div id="uploadbtn" class="button" x="140" y="10" width="60" height="30" text="Upload" font-size="10"></div> </div> <script src="../../build/pl.min.js"></script> <script> PL.GUI.getComponentById("uploadbtn").bindEvent('click', function(){ PL.loadTagValues(); }); </script> </body> </html> </pre>



```
});  
</script>  
</body>  
</html>
```



4 Часто задаваемые вопросы

Вопрос 1: «Как обрабатывать ошибки работы с устройством?»

Ответ 1: Разработчик может самостоятельно отлавливать исключительные ситуации при работе с тегом модуля PL302. Для этих целей вводится дополнительное событие “error”. Функция вызова с 2 параметрами (имя тега, тип ошибки). Доступные типы ошибок: **NetworkException**, **TagException**, **WrongValueException**.

Пример:

```
var bar = PL.GUI.getComponentById('bar');
bar.bindEvent('error', function(tag, errorName){
    console.log('Ошибка ', errorName, ' [' + tag + ']');
});
```

Для упрощения обработки ошибок от группы компонентов можно использовать функцию **PL.handleExceptions(callback)**. Обязательными аргументами функции обратного вызова являются тип ошибки и массив имен тегов.

Пример:

```
PL.handleExceptions(function(type, array){
    console.log(type, array);
});
```

Вопрос 2: «Иногда появляется модальное окно ошибки работы с тегами устройства. Как это исправить?»

Ответ 2: По умолчанию, при любых сбоях в системе коммуникации с устройством, фреймворк незамедлительно отображает модальное диалоговое окно, которое невозможно закрыть вручную (диалоговое окно закрывается при условии возобновления связи с устройством).

Пользователь обязан проверить наличие связи с устройством (возможны перебои в работе сети связи) и корректность имен тегов в HTML странице и JS скриптах. Если предыдущий этап не решил проблему, воспользуйтесь дополнительными глобальными переменными (объявляются до подключения js скрипта запуска фреймворка):

HIDE_ERROR_DIALOG – скрывать модальное диалоговое окно ошибки работы с тегами устройства (по умолчанию false).

AJAX_TIMEOUT – максимальное время ожидания ответа по AJAX в мс. (по умолчанию 1000 мс.).

AJAX_RETRIES – максимальное количество попыток запроса тега (по умолчанию 3).

MAX_GET_TAGS_COUNT – максимальное количество тегов, отправляемых в запросе на получение значений тега (по умолчанию 10).

MAX_SET_TAGS_COUNT – максимальное количество тегов, отправляемых в запросе на установку значений тега (по умолчанию 4).

Пример:

```
<script>
    HIDE_ERROR_DIALOG = true;
    AJAX_TIMEOUT = 1500;
    AJAX_RETRIES = 5;
```



```
MAX_GET_TAGS_COUNT = 8;  
MAX_SET_TAGS_COUNT = 2;  
</script>  
<script src="pl.min.js"></script>
```

Вопрос 3: «Используя нестандартное значение **timeout** почти для всех компонентов и каждый раз приходится добавлять вручную. Как поменять стандартное значение **timeout**?»

Ответ 3: Можно воспользоваться глобальной переменной **DEFAULT_TIMEOUT** (использование смотрите в ответе на вопрос 2), которая позволяет изменить стандартное время опроса тегов на заданное (по умолчанию 1000 мс). Если время опроса компонента не отличается от **DEFAULT_TIMEOUT**, то его объявление можно опустить.

Вопрос 4: «Если я правильно понимаю, то выводимый текст определяется блоком `data={'Работает': [51,100], 'Выключено': [0, 50]}`?»

Ответ 4: Компонент **Label** (п.п. 3.21 стр. 66) позволяет разработчику отображать значение тега устройства PL302. Выводимое значение может приводиться к необходимому типу, если указать атрибут **type** (integer, float; стр. 66). Указание специального атрибута **data** (стр. 68) в компоненте **Label** позволяет использовать интервальные значения тега. Так, например, если атрибут `data={'Работает': [51,100], 'Выключено': [0, 50]}`, то текст компонента автоматически заменяется на «Работает», если значение тега находится в пределах от 51 до 100 включительно и на «Выключено» при условии что значение тега находится в пределах от 0 до 50 включительно.

Вопрос 5: «К чему относятся 'default': 'Местный' или 'default': 'Дистанционно'? Что такое 'default'?»

Ответ 5: В компоненте **Label** атрибут **default** (стр. 68) позволяет задавать значение текста компонента по умолчанию при запуске фреймворка (данный текст отображается до первого запроса к устройству). В совокупности с заполненным атрибутом **data** (стр. 68), значение атрибута **default** автоматически подставляется в текст компонента, если текущее значение не соответствует ни одному из интервальных значений. Например, имеется следующий html код компонента:

```
<div class="label" x=100 y=100 name="A13" default="ТЕСТ" type="integer"  
timeout="1000" data={'Бумовая маска сработала': [53,53], "1": [0,50]}>  
</div>
```

Если текущее значение компонента «53», то будет отображен текст **«Бумовая маска сработала»**, если текущее значение «25», то будет отображено **«1»**, но если текущее значение отличается от текущих интервалов, например «152», то будет отображен текст **«ТЕСТ»**.

Вопрос 6: «Зачем создавать доп. метку через функцию `createLabel`, если у нас уже все метки прописаны через `div`-ы и их свойства задаются через `getComponentById().setState`?»

Ответ 6: Рассматриваемый фреймворк использует гибридный подход к созданию объектов, что позволяет создавать компоненты не только в HTML коде, но и в коде JS.

Вопрос 7: «Можно ли на лету генерировать страницу при помощи javascript, если большая часть компонентов типовые? (Учитывая то, что тратится много



времени на задание координат, размеров, отступов компонентов, логично было бы как-то автоматизировать эту работу.)”

Ответ 7: Фреймворк позволяет создание компонентов из JS скрипта на лету. К сожалению, избавиться от задания координат невозможно, так как фреймворк использует масштабируемую SVG графику (позволяет без потери качества отображать UI на всех устройствах одинаково) с виртуальной системой координат. Для автоматизации данного процесса ведется разработка полноценной визуальной среды разработки (для уточнения даты выхода необходимо связаться с руководством компании).

Вопрос 8: *“В предыдущей версии у вас использовалось 2 файла: ajax.js - общие функции получения данных, ту.js - пользовательские функции для отображения. Мне не понятно, как теперь реализуется получение данных с PL302 в новом фреймворке. Просьба пояснить, как интегрировать ajax.js в ваш фреймворк.”*

Ответ 8: Текущая версия фреймворка включает интегрированный ajax.js. Фреймворк самостоятельно формирует очередь запросов и производит их оптимизацию. Для работы с тегами вручную можно воспользоваться встроенным интерфейсом (Ответ 2, Вариант 2).

Если для личных целей разработчика необходим механизм AJAX, он может воспользоваться им так (фреймворк позволяет использовать jQuery без внешних зависимостей):

```
<script src="pl.min.js"></script>
<script>
    $.ajax(....);
</script>
```

Не рекомендуется работать с устройством напрямую по AJAX, так как ваши действия могут привести к неправильной обработке со стороны устройства (превышение одновременно устанавливаемых значений тега или их чтения и т.п.).

Вопрос 9: *“Что за типы данных указаны в div-ax? Типы integer, float не относятся к типам javascript и не описаны в вашей документации.”*

Ответ 9: В JS предусмотрен только один числовой тип (number). Но PL302 использует гораздо большее число типов переменных. Для упрощения работы фреймворк автоматически валидирует принятые значения тега преобразуя тип (в некоторых задачах нет необходимости отображать дробные значения, например, 23.000343). Для реализации используются функции из JS **parseFloat**, **parseInt**. По умолчанию в большинстве компонентов используется тип float (parseFloat).

Вопрос 9: *“Что за типы данных указаны в div-ax? Типы integer, float не относятся к типам javascript и не описаны в вашей документации.”*

Ответ 9: В JS предусмотрен только один числовой тип (number). Но PL302 использует гораздо большее число типов переменных. Для упрощения работы фреймворк автоматически валидирует принятые значения тега преобразуя тип (в некоторых задачах нет необходимости отображать дробные значения, например, 23.000343). Для реализации используются функции из JS **parseFloat**, **parseInt**. По умолчанию в большинстве компонентов используется тип float (parseFloat).

Вопрос 10: *“Как мне задавать различные тексты при использовании битовых масок, когда сочетание битов определяет выводимый текст?”*



Ответ 10: Для данной задачи можно использовать не только готовые компоненты Label и Text, но и любые доступные HTML объекты (при условии, что разработчик самостоятельно описывает адаптивный дизайн).

Вариант 1 (Label HTML): Для работы с битовыми масками рекомендуется использовать **type="integer"**, чтобы избежать ошибок при получении необходимого бита. Битовая маска преобразуется в десятичное число, например, битовая маска $(110101)_2 = (53)_{10}$ (для удобства можно воспользоваться калькулятором или онлайн ресурсом, например, <http://www.binaryhexconverter.com/binary-to-decimal-converter>). Воспользовавшись вышеизложенным ответом 1 используем интервальные значения тега, таким образом, чтобы при изменении значения тега на "53" текст менялся на значение "Битовая маска сработала", то есть атрибут `data="{\"Битовая маска сработала\": [53,53]}`. Полный код примера:

```
<div class="label" x=100 y=100 name="A13" type="integer" timeout="1000" data="{\"Битовая маска сработала\": [53,53]}\">
</div>
```

Все остальные возможные значения битовой маски также необходимо перевести в десятичное значение и добавить в атрибут **data** через запятую, например:

```
<div class="label" x=100 y=100 name="A13" type="integer" timeout="1000" data="{\"Битовая маска сработала\": [53,53], \"1\": [54,54], \"2\": [55,55]}\">
</div>
```

Вариант 2 (Text/HTML JS): Для работы с тегами в JS коде вручную, фреймворк предусматривает несколько вариантов использования. Если в Вашем коде есть в наличии компонент, с необходимым тегом, то необходимо привязать к нему обработчик события **value**, и производить обработку в JS коде. Если же в вашем HTML документе нет ни одного компонента, связанного с данным тегом — необходимо использовать интерфейсы `getTagValue` или `getBoolTagValue`. Разработчик может производить самостоятельную обработку и менять любой компонент или их совокупность. Для получения **n**-ого бита из числа **v**, необходимо написать: $(v \gg (n - 1)) \& 1$. Приведем пример:

```
v = 34;
n = 2 (2 бит, начиная отсчет с 1);
(v >> (2 - 1)) = (34 >> 1) = 17 ;
17 & 1 = (10001) & (1) = 1
```