Dive Into Shortcodes

Software Engineer

Little Rock, AR

The Crossing, EPC

Shortcode Demos!

Agenda

1. Introducing Shortcode Syntax
2. Designing a Dashboard
   - KPI Shortcode
   - Trend Chart Shortcode
   - Panel Shortcode
   - Chart Shortcode
3. Designing a Public Webpage
   - Google Maps Shortcode
   - Media Player Shortcode
   - Scheduled Content Shortcode
   - Scripturize Shortcode
4. Designing Your Own Shortcode

Shortcodes are predefined sections of Lava, HTML, and CSS

They are particularly useful for

- ## Frequently Repeated Lava

  Any time you find yourself writing the exact same Lava over and over you could likely turn it into a reusable Shortcode

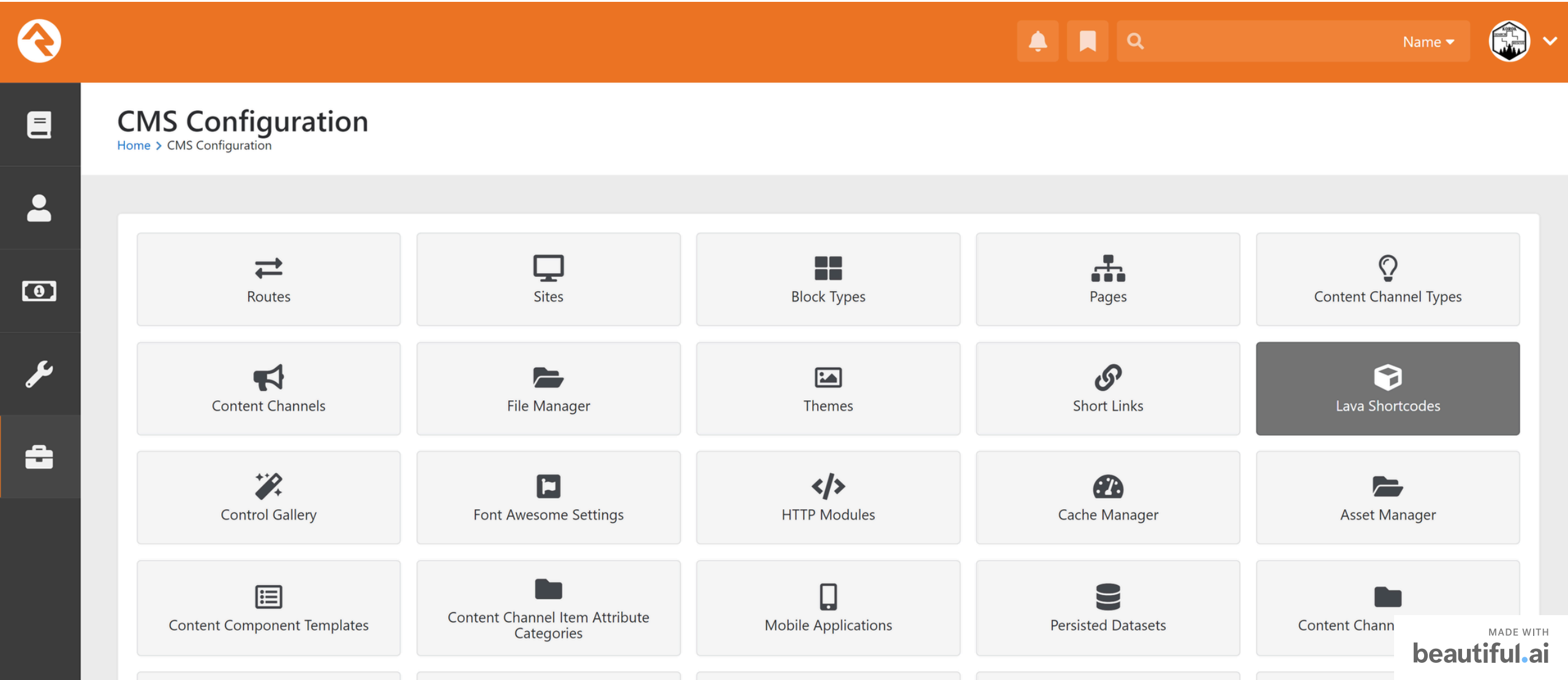
- ## Making a complex design easy for ministry staff to use

  Expand the elements your ministry staff can add to page by putting the complex code into a Lava Shortcode and giving them something accessible

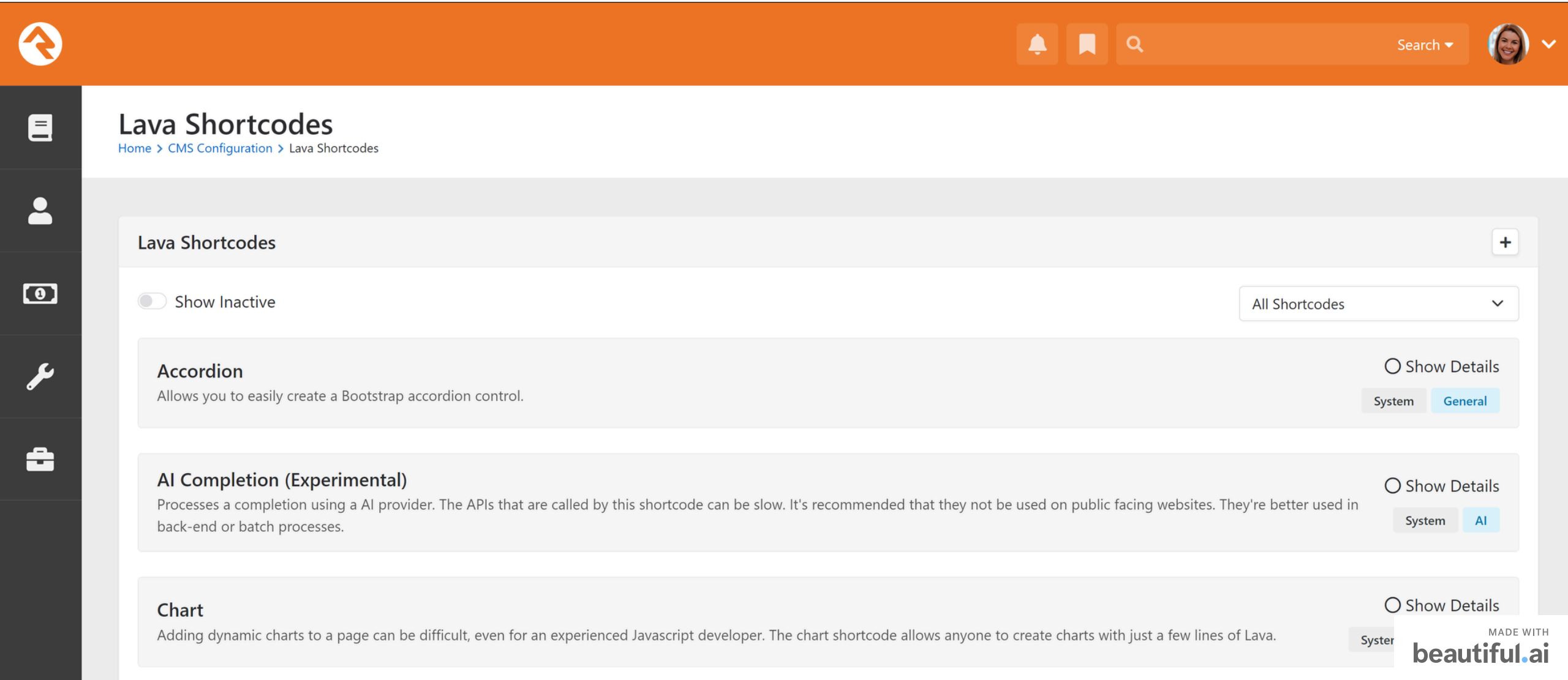
- ## Universal Changes

  If you have a shared template for your elements it is easy to make global changes to them by only editing the Lava Shortcode

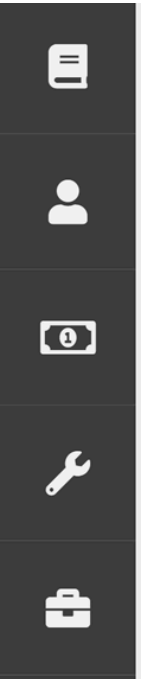
Shortcodes live under the CMS Configuration



The shortcodes page lists all available shortcodes



Clicking on a shortcode displays all configuration options



KPI

Create quick key performance indicators.

Show Details

System Reporting

Basic Usage:

```
{[kpis]}
  [[ kpi icon:'fa-highlighter' value:'4' label:'Highlighters' color:'yellow-700']][[ endkpi ]]
  [[ kpi icon:'fa-pen-fancy' value:'8' label:'Pens' color:'indigo-700']][[ endkpi ]]
  [[ kpi icon:'fa-pencil-alt' value:'15' label:'Pencils' color:'green-600']][[ endkpi ]]
{[endkpis]}
```

Shortcode Options

- **title** (blank) – An optional main title for the group of KPIs.
- **subtitle** (blank) – The smaller body sized text to show under the title. Only shown if provided.
- **showtitleseparator** (true) – The label for the KPI.
- **columncount** (4) – The number of columns to display at the large responsive size. Will decrement by one for each subsequent breakpoint (lg: 4, md: 3, sm: 2, xs: 1).
- **columnmin** (blank) – The minimum width for a column to display, used to force columns to wrap onto the next line when out of space.
- **size** (md) – The size of the KPI, default is medium. Options include `sm, lg, xl`.
- **style**(card) - The display style of the KPI. Options include `card` and `edgeless`.
  - **default** – Default KPI with icon, and no border.
  - **card** – KPI with card like appearance and border.
- **iconbackground** (true) - Set the value to true to display a background behind the FontAwesome icon.
- **tooltipdelay** (0) - Milliseconds to delay showing and hiding the KPI description tooltip (if one is provided).

KPI Options

- **icon** (blank) – The class of the FontAwesome icon, no icon will display if left blank.

Shortcodes are defined with curly braces and brackets

Similar to Lava Tags, Shortcodes can be inline or block.

Inline:

```
{[ shortcodename ]}
```

Block:

```
{[ blockshortcodename ]}
    ...Inner Contents...
{[ endblockshortcodename ]}
```

Block Shortcodes can have child items

These child items are defined by double brackets.

```
{[ blockshortcodename ]}
    [[ childitem ]]
        ...Child Item Content...
    [[ endchilditem ]]
{[ endblockshortcodename ]}
```

Simple Stats

Designing a Dashboard

RX 2024 Shortcodes

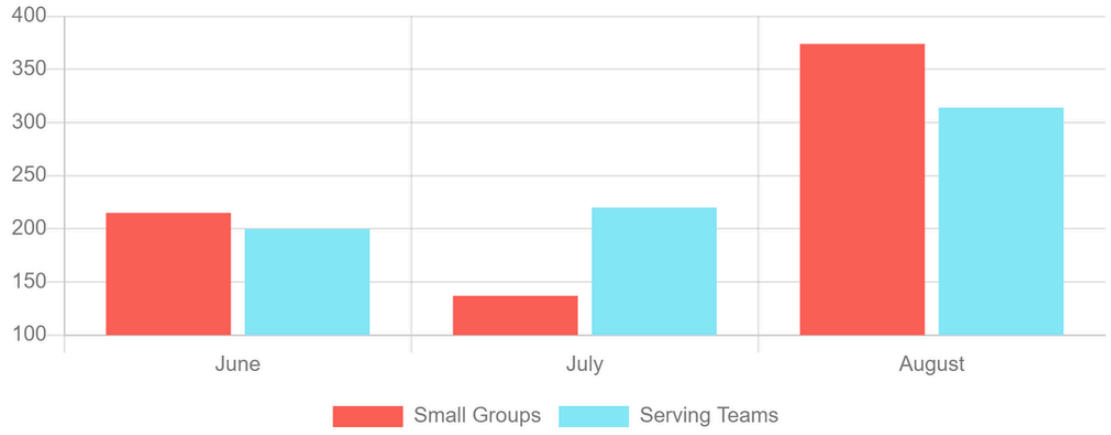
4,324
Service Attendance

$25,000 +4%
Giving

47
Sermon Watches
Higher than last week

YTD Giving

Group Attendance Last 3 Months



Attendance Growth: Our church attendance has grown by 300% since last year! We're now hosting services in three parallel dimensions simultaneously.
Miracle Count: Witnessed 50 miraculous healings last month, including the regrowth of a missing sock after a prayer circle.
Sermon Impact: 99% of attendees reported achieving instant enlightenment during last Sunday's sermon, with one member achieving levitation.
Community Engagement: Our bake sale raised enough funds to build a spaceship for outreach to neighboring galaxies.
Prayer Power: Our prayer group's collective energy is now capable of temporarily altering the Earth's rotation speed, adding an extra hour to every Sunday service.
Divine Downloads: Members received 100 terabytes of spiritual wisdom during our latest retreat, enough to fill the Library of Congress 10 times over.

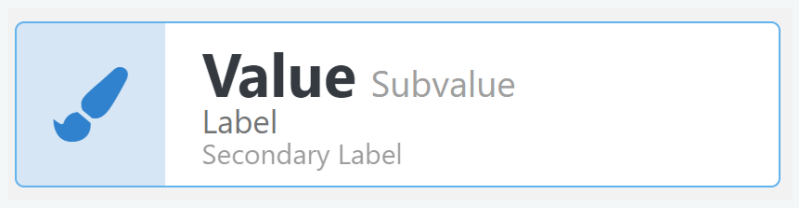
The KPI Shortcode makes simple stats stand out

Type: Block

Markup:

```
{[ kpis ]}
  [[ kpi icon:'fa-paint-brush' value:'Value' color:'blue-600' ]][[ endkpi ]]
{[ endkpis ]}
```

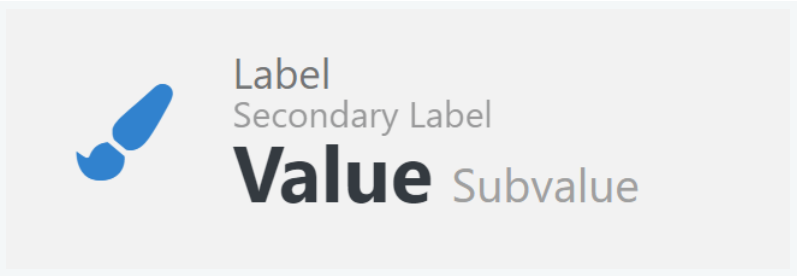
- **icon:** a font awesome icon class
- **value:** the statistic you are trying to convey
- **color:** the color for the icon



Get a flat look by removing the card border and icon background

```
{[ kpis style:'edgeless' iconbackground:'false' ]}
  [[ kpi labellocation:'top' ]][[ endkpi ]]
{[ endkpis ]}
```

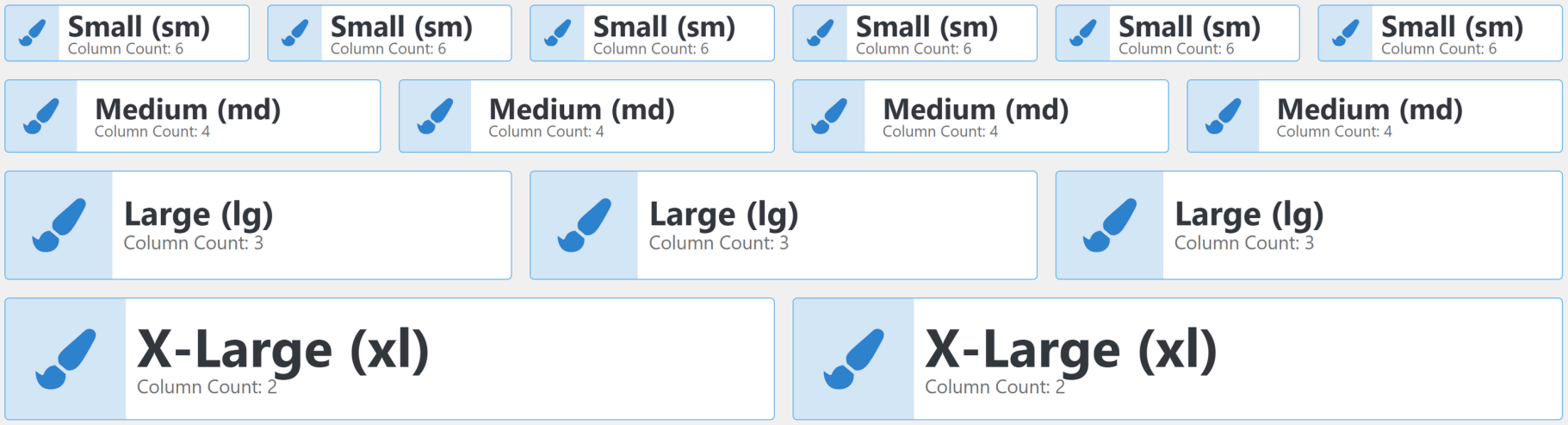
Great for KPIs that are inside another element like a card, well, or panel.



Easy updates to KPI width and height

Size and Column Count are set in the **kpis** tag for all **kpi** items inside them. Manual overrides are also available.

```
{[ kpis size:'sm' columncount:'6' ]}
    [[ kpi icon:'fa-paint-brush' value:'Small (sm)' label:'Column Count: 6' color:'blue-600' ]][[
endkpi ]]
{[ endkpis ]}
```



KPI Colors

KPIs have 10 simple color options with 9 different shades, you can also use any CSS Hex Value

There are three parts to Font Awesome icons

- ## Icon Type
  The parent class that defines the library of font awesome icons to use, the default is "fa" for regular font awesome icons.
  Some brand icons live in the "fab" library.

- ## Icon Name
  The name of the specific icon, like "paint-brush"

- ## Icon Class
  To display a specific icon you need to first list the class name for the icon type. Then you will write the icon name after "fa-"
  fa fa-paint-brush

KPIs accept the icon type class in the kpis tag and the icon class in the kpi items

```
{[ kpis icontype:'fa-duotone' ]}
    [[ kpi icon:'fa-paint-brush' ]]
{[ endkpis ]}
```

Not all icon type libraries for Font Awesome are free.

By default Rock only has the free libraries so you won't have access to pro libraries like "fa-duotone" which displays icons shaded with two tones unless you purchase and install them on your Rock instance.

Panels create simple visual boundaries

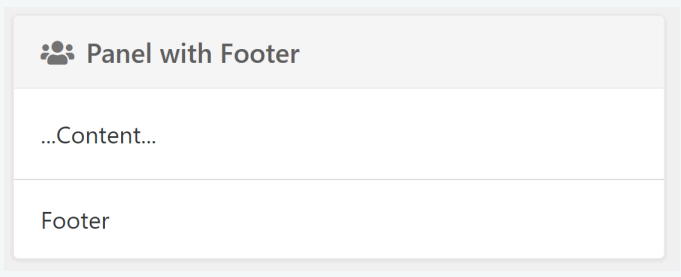
Type: Block

Markup:

```
{[ panel icon:'fa fa-users' title:'User Information' ]}
    ...Content...
{[ endpanel]}
```

- **icon:** a font awesome icon class, unlike kpis you will write the full class here
- **title:** the name of this section of content
- **content:** what will display inside the panel

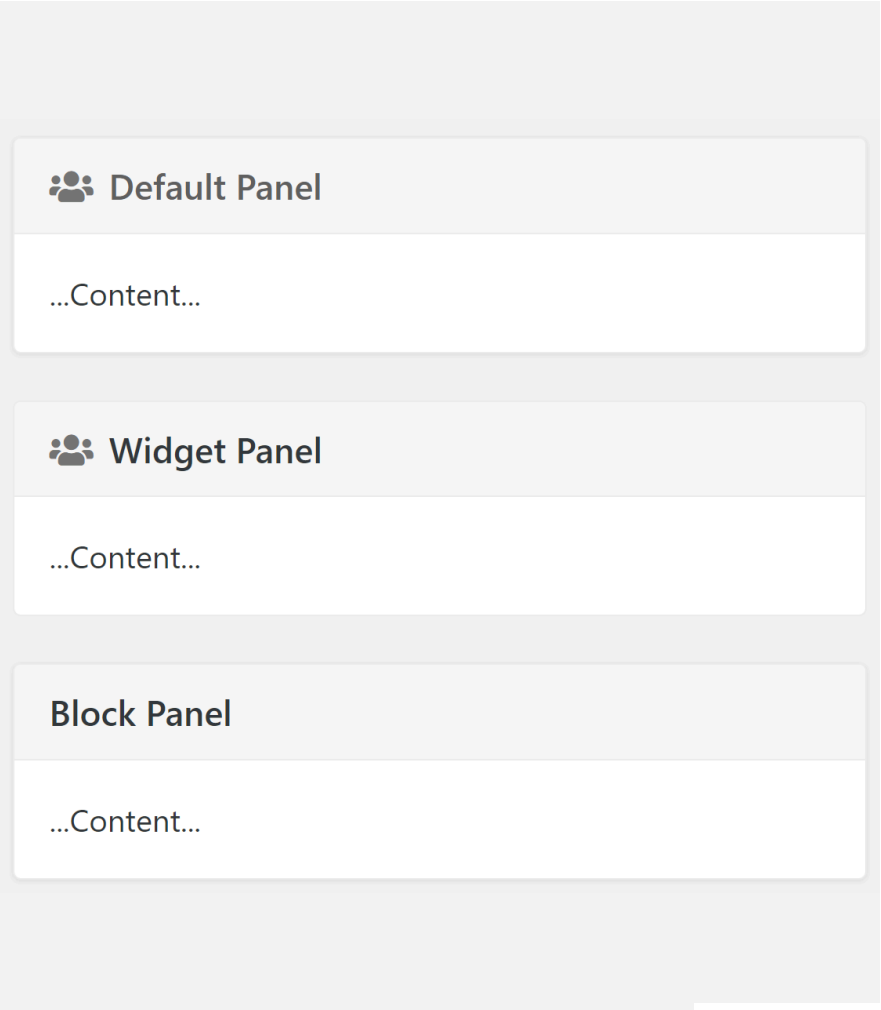
Technically nothing is required, but an empty panel tag will result in a white box on the page.



Panel types can change the way a panel is displayed

```
{[ panel type:'default' ]}
    ...Content...
{[ endpanel]}
```

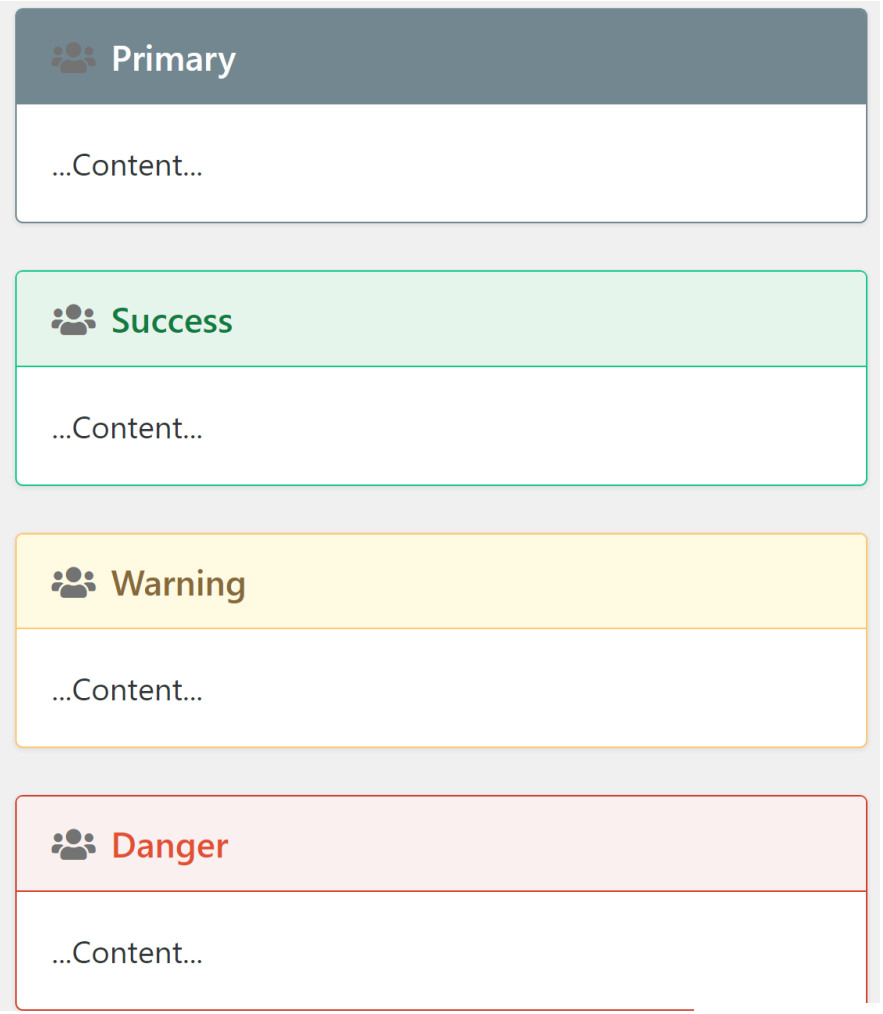
- **default:** the default option, displays a panel as you usually see it
- **widget:** removes the border and shadow to create a flat element
- **block:** similar to default but icons will be hidden and many block panels in a bootstrap grid will stack differently



Panel types can change a panel's color

```
{[ panel type:'primary' ]}
    ...Content...
{[ endpanel]}
```

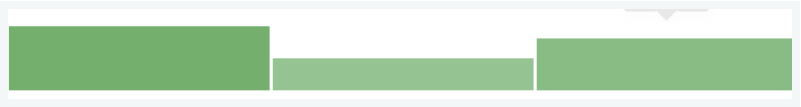
- **primary:** your primary brand color from your theme
- **success:** same light green as success alerts
- **warning:** same light yellow as warning alerts
- **danger:** same red coloring as danger alerts



Trend Charts provide a clear visual without the extra clutter of a full chart

Type: Block

Markup:



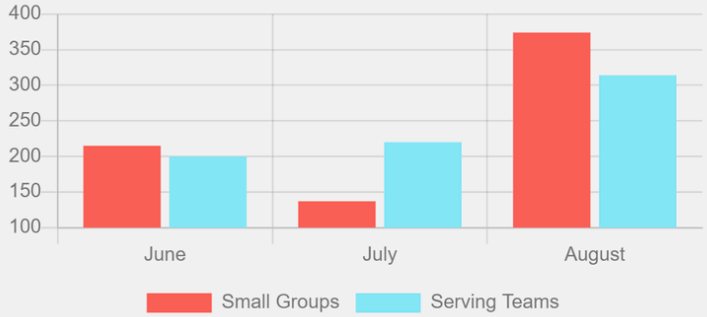
```
{[ trendchart ]}
    [[ dataitem label:'Label' value:'100' color:'#74AF6E' ]][[ enddataitem ]]
{[ endtrendchart ]}
```

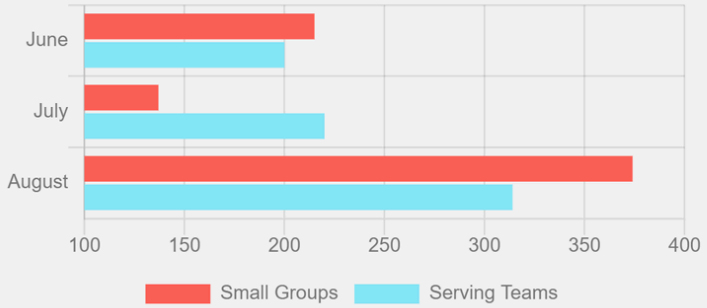
- **label:** will appear as a tooltip when a user hovers over the bar
- **value:** a numeric value to determine the height of the bar, it is all relative to your highest value being 100% height and all other values the percentage they are of the highest value
- **color:** a hex value that will provide the color for the bar

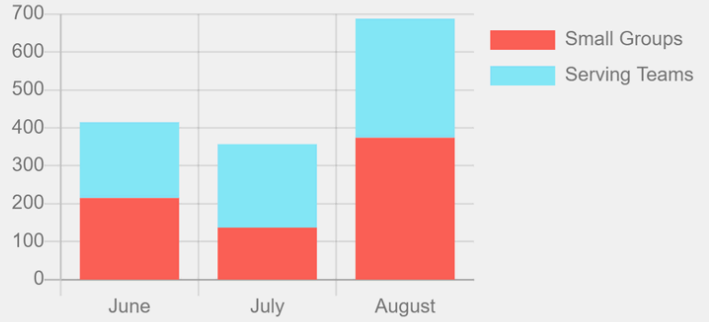
A Trend Chart is Actually a Stylized HTML List

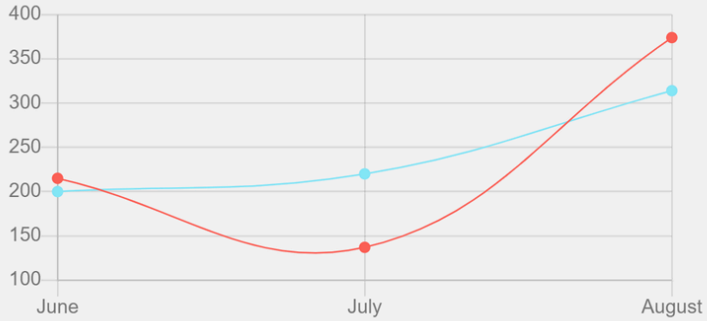
```
<ul class="trend-chart">
  <li><span style="backgorund: lightgreen; height: 100%;"></span></li>
</ul>
```

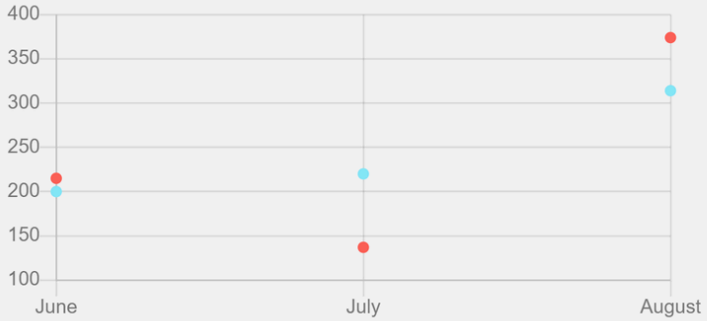
Charts are powerful and diverse data visualization tools

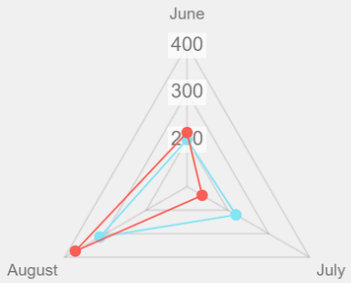


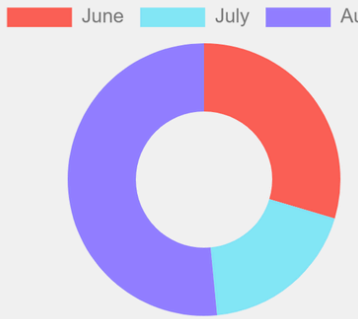


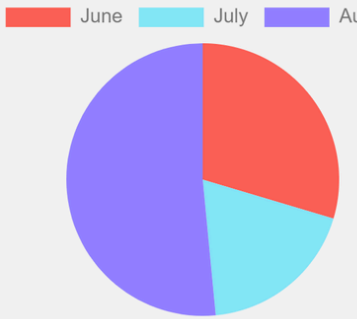


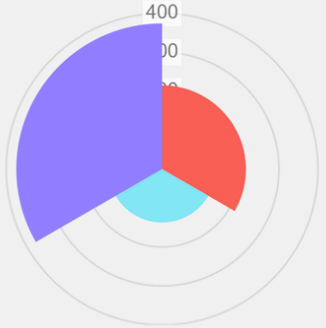












Basic chart configuration requires

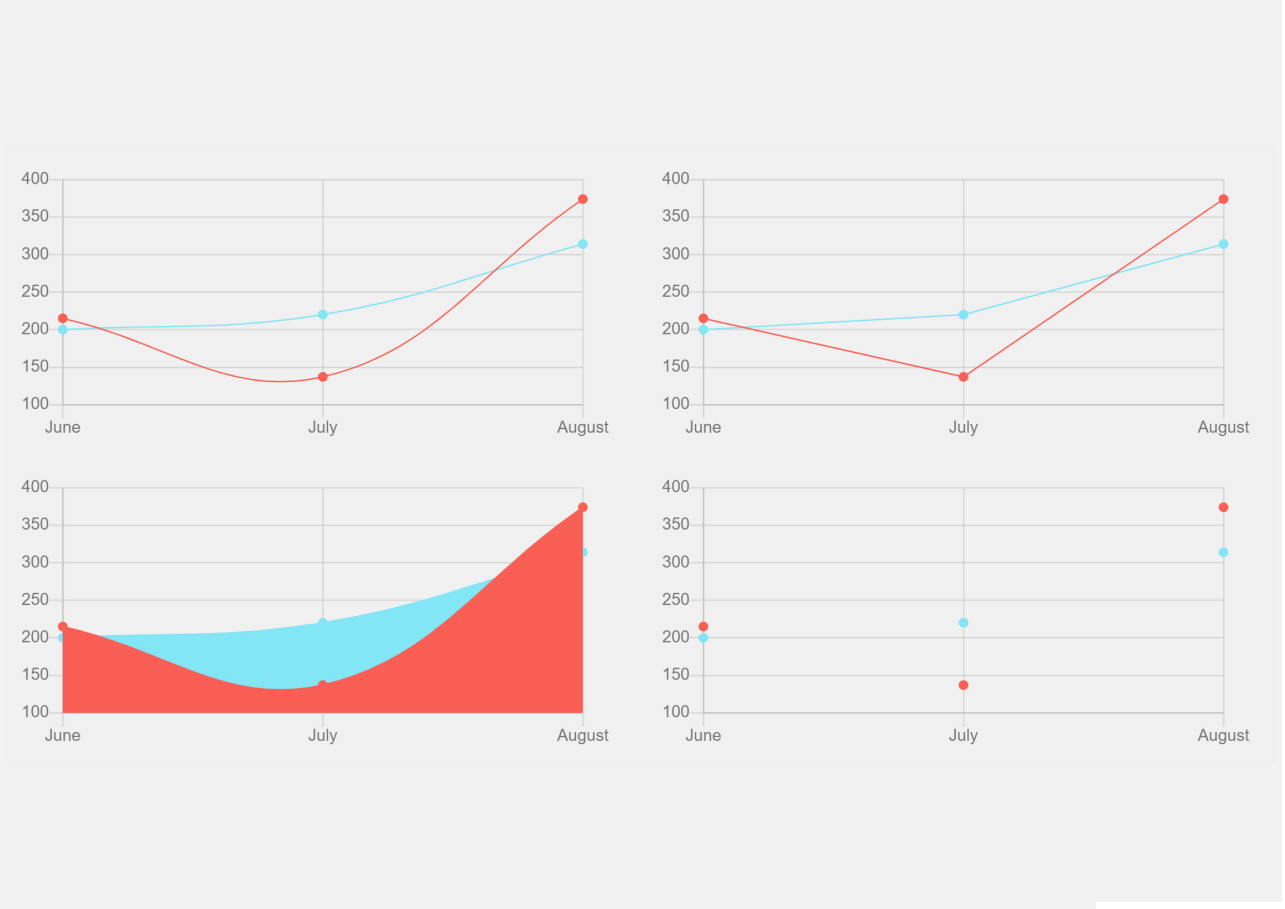
1. **Type:** the type of chart to display (bar, line, pie, etc.)
2. **Labels:** information about the data sets and the values they are being assigned
3. **Data:** numeric information about the data
4. **Color:** information about how this data should be colored on the chart

There are three types of bar chart you can use

- **bar:** a regular bar chart
- **horizontalBar:** a bar chart oriented horizontally
- **stackedbar:** data stacked together to show comprehensive values

There is one type of line chart that can be customized in many ways

- **curvedlines** (true): the default setting
- **curvedlines** (false): lines go directly point to point
- **filllinearea** (true): fills the space under the line
- **bordercolor:** using an rgba with 0 for alpha (transparency) turns a line chart into a scatter plot



Charts with singular configurations

- **doughnut:** a pie chart without the center
- **pie:** basic pie chart
- **radar:** for complex comparisons with multiple criteria for data
- **polarArea:** radar meets pie

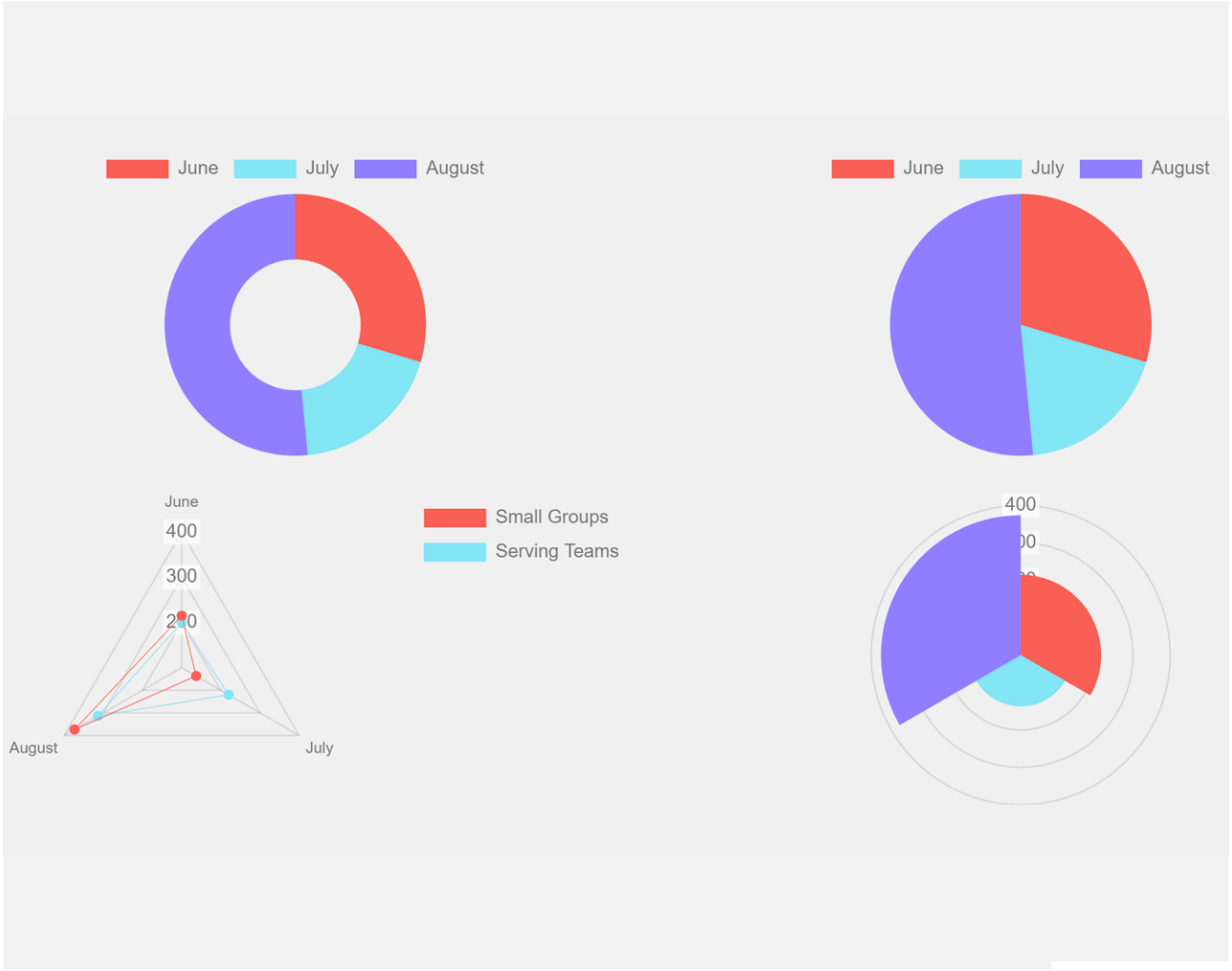


Chart data can be configured two ways

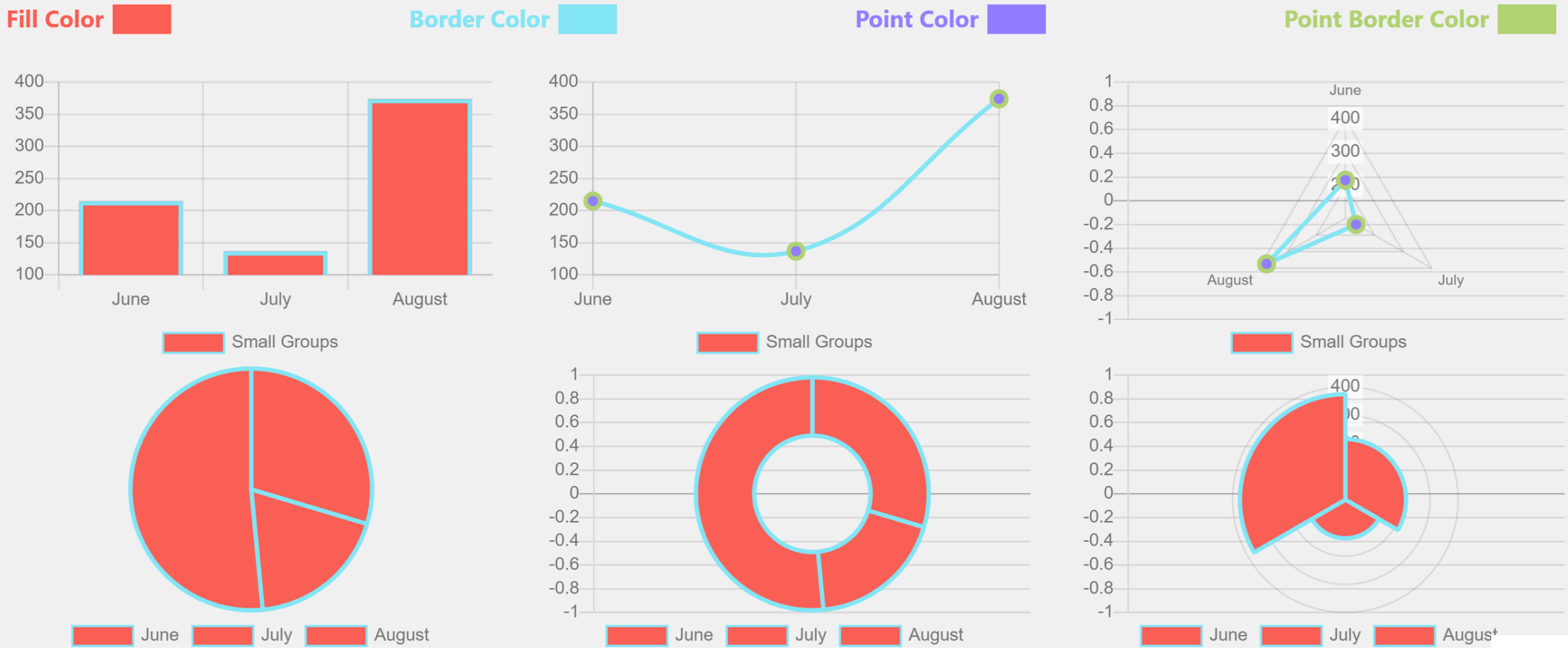
- **dataitem:** for single data sets (***label*** and ***value*** on the dataitem)
  Comparing attendance for serving teams each month of the summer

```
{[ chart type:'bar' ]}
  [[ dataitem label:'June' value:'200' ]] [[ enddataitem ]]
  [[ dataitem label:'July' value:'220' ]] [[ enddataitem ]]
{[ endchart ]}
```

- **dataset:** for multiple data sets (***labels*** on the chart for axis, ***label*** and ***data*** on the dataset)
  Comparing attendance for serving teams and small groups

```
{[ chart type:'bar' labels:'June, July, August' ]}
  [[ dataset label:'Small Groups' data:'215, 137, 374']] [[ enddataset ]]
  [[ dataset label:'Serving Teams' data:'200, 220, 314' ]] [[ enddataset ]]
{[ endchart ]}
```

Chart colors are configured differently depending on the type of chart



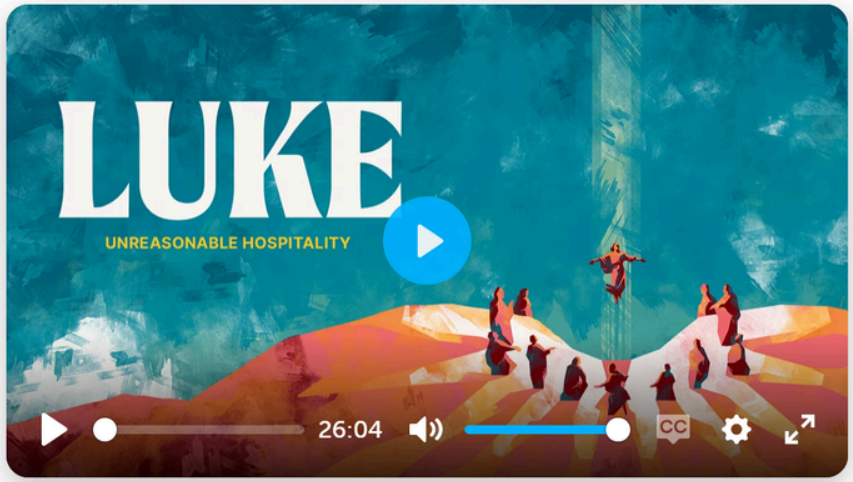
CSS Hex Colors, RGB, RGBA, and Named Colors are recognized

Other useful configuration options

- **legendshow:** the option to show the dataset labels in a legend
- **legendposition:** where the legend should be displayed
- **chartheight/chartwidth:** the height/width of the chart in pixels
- **valueformat:** is your data currency? percentages?
- **xaxisshow/yaxisshow:** the option to hide the x or y axes
  pie charts hide the graph axes by default, doughnut, polarArea, and radar do not
- **xaxistype:** linear by default, can be set to time when dealing with data on specific dates, or a fixed 0-100 scale
  The time option does not work well with bar charts

Activity Tracking

Designing a Public Webpage



Last Sunday...

Humility and Gratitude

Dave teaches from Luke 17 about how we experience restoration as we obey the commands of Jesus.

Explore Past Sermons

Join us at 8:30, 9:45, and 11

3615 Southland Dr
Columbia MO, 65203

Scheduled Content

Type: Block

Markup:

```
{[ scheduledcontent scheduleid:'123' ]}
    ...Content...
{[ endscheduledcontent ]}
```

- **scheduleid:** the id of the schedule you want to use
- **schedulecategoryid:** when you need multiple schedules, use the category id instead of the schedule id

Other Configuration Options

- **showwhen:** live, notlive, both

  Should the content display during the schedule, any time not during the schedule, or both

- **roleid**

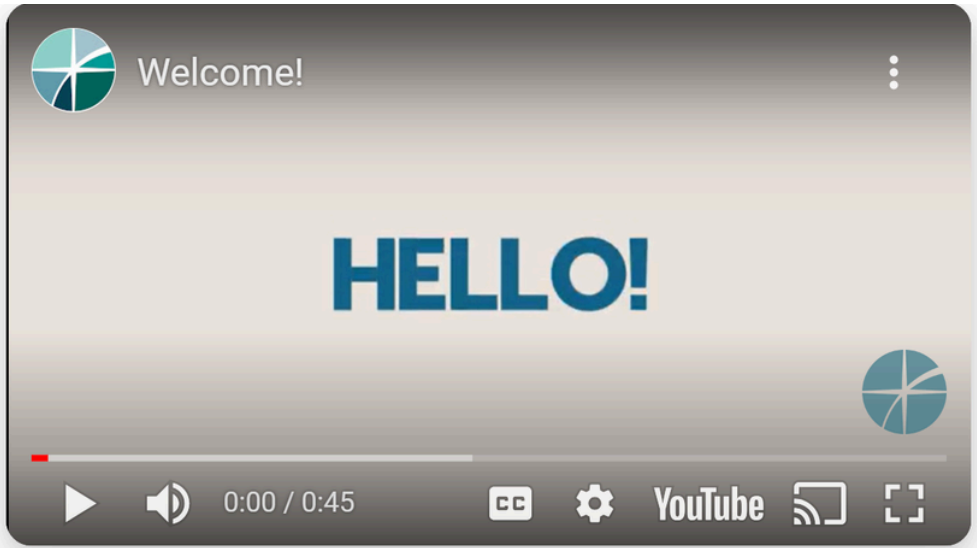
  The id of any group to limit who the content should be visible to

- **lookaheaddays:** 30

  The number of days we should look ahead to find the next occurrence of the schedule

Conditional Rendering Based on Schedule

```
{[ scheduledcontent scheduleid:'2729' ]}
  {[ mediaplayer media:'113' ]}{[ endmediaplayer ]}
{[ endscheduledcontent ]}
{[ scheduledcontent scheduleid:'2729' showwhen:'notlive' ]}
  {[ mediaplayer media:'112' ]}{[ endmediaplayer ]}
{[ endscheduledcontent ]}
```



Can't make it in person?

Watch the Livestream

Scheduled Content has Merge Fields Available

```
{[ scheduledcontent scheduleid:'1234' showwhen:'both' ]}
    {% if IsLive == true %}
        {[ mediaplayer media:'113' ]}{[ endmediaplayer ]}
        Service ends at {{ OccurrenceEndDateTime | 'st' }}
    {% else %}
        Join us on {{ NextOccurrenceDateTime | Date:'dddd' }}!
    {% endif %}
{[ endscheduledcontent ]}
```

- **IsLive:** true/false for if the provided schedule is currently active
- **OccurrenceEndDateTime:** When the current schedule will end
- **NextOccurrenceDateTime:** When the next occurrence will start

Media Player

Type: Block

Markup:

```
{[ mediaplayer media:'112' ]}{[ endmediaplayer ]}
```

- **media:** the id of the item in one of your Media Accounts

```
{[ mediaplayer src:'https://www.youtube.com/watch?v=dQw4w9WgXcQ' ]}{[ endmediaplayer ]}
```

- **src:** the url of the media to display

Player Controls

- **autoplay:** Determines if the media should play automatically, browsers usually prevent this unless audio is muted
- **hidecontrols:** Will hide the controls after two seconds of inactivity
- **volume:** 0 to 1 level of volume with 1 being 100%
- **controls:** The controls that should be available to the user

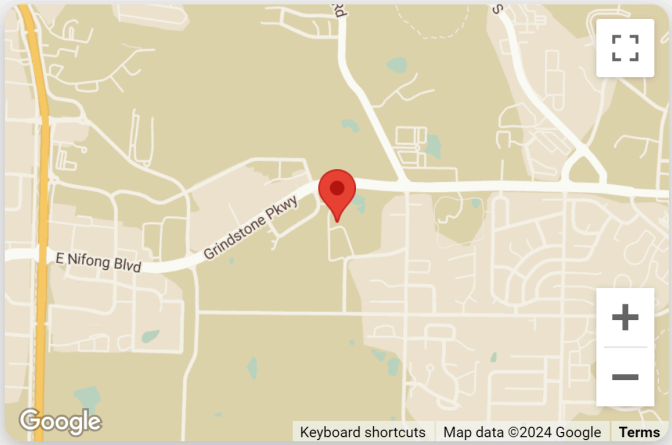
  play, restart, rewind, fast-forward, progress, mute, captions, pip (picture-in-picture for Safari), airplay (for Safari), download, fullscreen

The Order of Controls Will Determine How They are Displayed

Tracking Controls

- **autoresumeindays: (7)** The number of days to go back and pick up where a user left off watching
- **combineplaystatisticsindays: (7)** The number of days to go back and find an existing interaction for the current user
- **relatedentityid:** If the interaction for watching this media should be related to another entity you can provide the id for the interaction's related entity field

Google Maps



Type: Block

Markup:

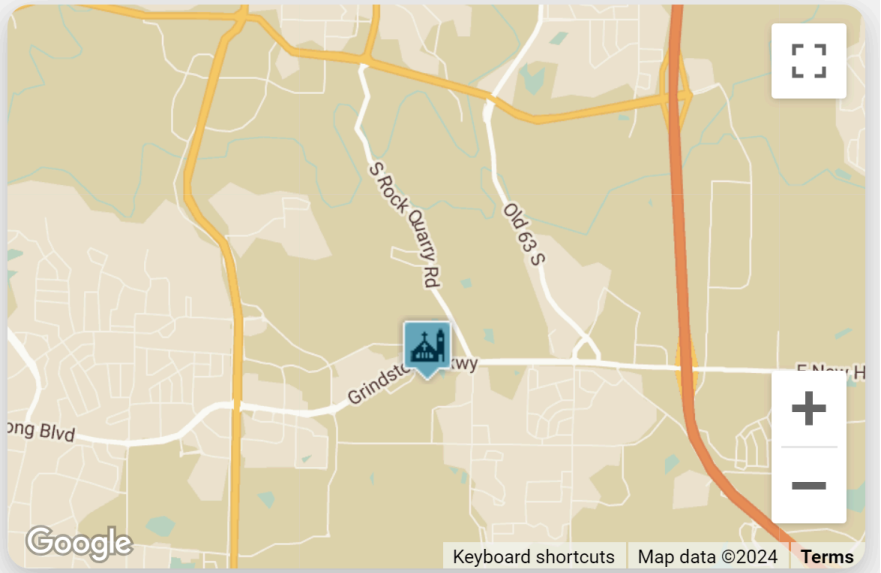
```
{[ googlemap ]}
    [[ marker location:'38.9116785,-92.3174238' title:'The Crossing']][[ endmarker ]]
{[ endgooglemap ]}
```

- **center :** the coordinates that should be the center of the map, unnecessary if a marker is used
- **zoom:** the level of zoom for the map
  1 = world > 5 = continent > 10 = city > 15 = streets > 20 = buildings

Map and Markers Can be Styled

```
{[ googlemap ]}
    [[ marker location:'38.9116785,-92.3174238'
       icon:'https://linkToMyIcon']][[ endmarker
]]
    [[ style ]]
        [ JSON For Map Style ]
    [[ endstyle ]]
{[ endgooglemap ]}
```

The Lava Shortcode documentation provides links to websites to generate custom map icons you can download and interfaces to generate the desired style JSON.



A Warning

Scripturize Shortcode

We All Abbreviate Scripture References Differently

With a focus on flexibility the scripturize shortcode will accept a variety of annotations for scripture references.

- John 3:16
- Jhn 3:16
- Jn 3:16
- Mark 4
- Mrk 4
- Mk 4

We Could Therefore Assume Luke Would Be

- Luke 4
- Lke 4
- Le 4
- Correct
- That's Nothing, Luk 4
- That's Leviticus 4

Some Potentially Problematic Abbreviations

- Rm 12
- Is 2
- Ps 4
- Room or Romans?
- Is or Isaiah?
- Post Script or Psalms?

Not as Hard as You Think

Designing Your Own Shortcode

The Basic Components You'll Need

- **Tag Name**
  What keyword do you want to use to access this shortcode?
- **Inline or Block**
  Will you allow inner content?
- **Markup**
  The code!
- **Parameters**
  Configuration options you wish to allow

Let's make a Bootstrap Carousel

Markup for Creating a Bootstrap Carousel

```
<div id="carouselExample" class="carousel slide" data-ride="carousel">
  <ol class="carousel-indicators">
    <li data-target="#carouselExample" data-slide-to="0" class="active"></li>
    <li data-target="#carouselExample" data-slide-to="1"></li>
  </ol>
  <div class="carousel-inner">
    <div class="carousel-item item active">
      <img class="d-block w-100" src="..." alt="First slide">
    </div>
    <div class="carousel-item item">
      <img class="d-block w-100" src="..." alt="Second slide">
    </div>
  </div>
</div>
```

Replace The Slide Contents With User Input

```
<div class="carousel-inner">
    {% for slide in slides %}
        <div class="carousel-item item {% if forloop.first == true %}active{% endif %}">
            <img class="d-block w-100" src="{{ slide.src }}" alt="{{ slide.title }}">
            <div class="carousel-caption d-none d-md-block">
                <p>{{ slide.content }}</p>
            </div>
        </div>
    {% endfor %}
</div>
```

```
[[ slide title:'Cat Slide' src:'https://placekitten.com/g/600/200' ]]This is a cat![[
endslide ]]
```

Replace The Slide Contents With User Input

```
<ol class="carousel-indicators">
    {% for slide in slides %}
        <li data-target="#carouselExample"
            data-slide-to="{{forloop.index0}}"
            {% if forloop.first == true %}class="active"{% endif %}
        ></li>
    {% endfor %}
</ol>
```

In this instance we don't need any input from the user to make the indicators. We just need to know how many slides they added.

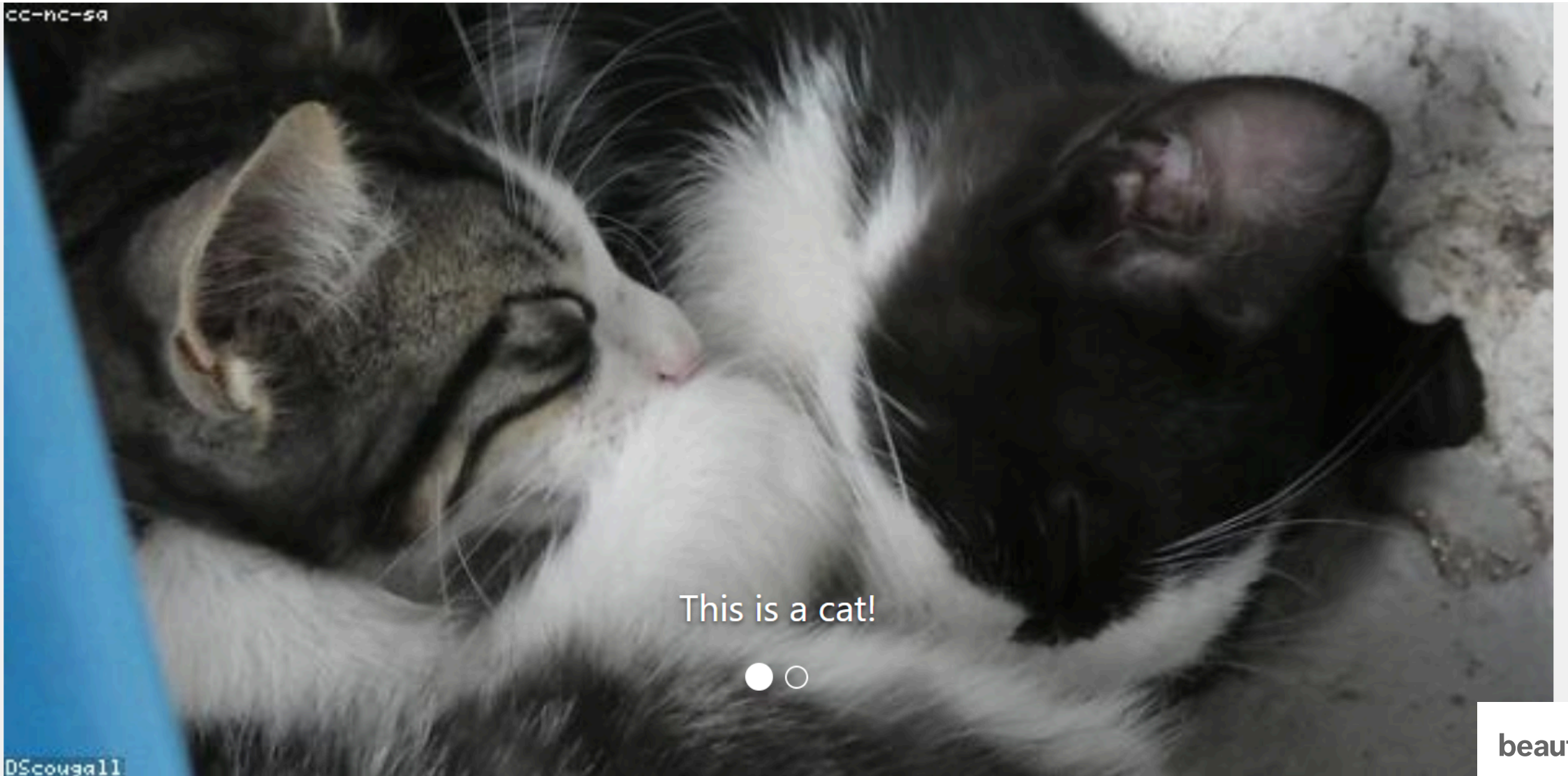
The Current Markup

```
<div id="carouselExample" class="carousel slide" data-ride="carousel">
  <ol class="carousel-indicators">
    {% for slide in slides %}
        <li data-target="#carouselExample" data-slide-to="{{forloop.index0}}"
            {% if forloop.first == true %}class="active"{% endif %}></li>
    {% endfor %}
  </ol>
  <div class="carousel-inner">
    {% for slide in slides %}
        <div class="carousel-item item {% if forloop.first == true %}active{% endif %}">
            <img class="d-block w-100" src="{{ slide.src }}" alt="{{ slide.title }}">
            <div class="carousel-caption d-none d-md-block"><p>{{ slide.content }}</p></div>
        </div>
    {% endfor %}
  </div>
</div>
<script> $(document).ready(() => { $('.carousel').carousel(); }); </script>
```

Using the Shortcode

```
{[ carousel ]}
    [[ slide title:'Cat Slide' src:'https://placekitten.com/g/600/200' ]]
        This is a cat!
    [[ endslide ]]
    [[ slide title:'Another Cat Slide' src:'https://placekitten.com/g/300/100' ]]
        Another cat!
    [[ endslide ]]
{[ endcarousel ]}
```

The Result is a Bootstrap Carousel



This Markup Breaks When We Add Multiple Carousels to a Page

Clicking the indicators for the second carousel slides the first!

This Markup Breaks When We Add Multiple Carousels to a Page

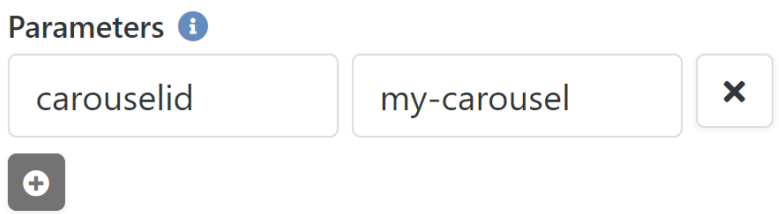
The markup we used has a hard coded id for the carousel element that is now used for multiple elements.

```
<div id="carouselExample" class="carousel slide" data-ride="carousel">
  <ol class="carousel-indicators">
    <li data-target="#carouselExample" data-slide-to="0" class="active"></li>
    <li data-target="#carouselExample" data-slide-to="1"></li>
  </ol>
```

Resolve Conflicts With a Parameter

The markup we used had a hard coded id for the carousel element that is now used for multiple elements.

```
<div id="{{ carouselid }}" class="carousel slide" data-ride="carousel">
  <ol class="carousel-indicators">
    {% for slide in slides %}
        <li data-target="#{{ carouselid }}" data-slide-to="{{forloop.index0}}"
            {% if forloop.first == true %}class="active"{% endif %}></li>
    {% endfor %}
  </ol>
```



Summary

- Shortcodes make complex HTML elements easily accessible
- There are hundreds of configuration options to customize Shortcodes for your specific use cases
- Shortcodes keep a unified source for complex elements so making global change is faster and simpler