

# Lightning Talk

M. PITCEL

APRIL 5, 2019

# Creating an Image Carousel

React Component that Cycles Specified Images

# What is a Carousel?

- ▶ Visual section on a webpage that displays Images
- ▶ Images will
  - ▶ Cycle automatically
  - ▶ Cycle via User clicking buttons
  - ▶ Be chosen individually by the User

# The Code

HOW TO SETUP AND USE THE  
COMPONENT IN REACT

# Utilizing a Component

- ▶ The “important” pieces to make the Carousel Work
  - ▶ Configuration Items to define which Images to use
  - ▶ Six Functions to change the images in **Carousel.jsx**
  - ▶ Style to make the images transition in **carousel.css**
  - ▶ Setup the Component on a page like **Home.jsx**
- ▶ We can look at all of these pieces

# Configuration Items

# Configuration Items

- ▶ For the Carousel to appear, two things must occur:
  1. Add images to the **theme** folder
  2. Add the list of want-to-display images to the **config.js** file at **home\_page\_carousel\_images: []**,

# Configuration Items

- ▶ Need a function in `getConfig.js` to get the info from `config.js`:

```
export function getCarouselImageNames() {  
    let nameList = '';  
    if (window.configruntime.gd3.home_page_carousel_images) {  
        nameList =  
            window.configruntime.gd3.home_page_carousel_images;  
    }  
    return nameList;  
}
```

# Six Functions

# Six Functions

- ▶ `componentDidMount()`
- ▶ `componentWillUnmount()`
- ▶ `selectImage(index)`
- ▶ `forwardImage()`
- ▶ `backwardImage()`
- ▶ `displayImage(imageIndex)`

# Six Functions (1)

```
componentDidMount() {  
  // Select the first image  
  this.displayImage(this.state.imageIndex);  
  // Start the Carousel transitions at page load  
  this.carouselInterval = setInterval(() => {  
    if (this.state.carouselActive) {  
      this.forwardImage();  
    } else {  
      this.setState({carouselActive: true});  
    }  
  }, 1500);  
}
```

- ▶ This function is utilized when the Component loads
- ▶ Set the Carousel to start at the first image using **displayImage(imageIndex)**
- ▶ Every 1.5 seconds, ensure that the carousel is active (**carouselActive: true**)

# Six Functions (2)

```
componentWillUnmount() {  
  // Clear the Carousel Interval when the User  
  // leaves this page  
  clearInterval(this.carouselInterval);  
}
```

- ▶ This function is utilized when the User leaves the page
- ▶ The carousel loop is cleared to ensure that memory is not utilized in the background when not necessary

# Six Functions (3)

```
selectImage(index) {  
    // User clicks to select a specific Image  
    this.setState(  
        {carouselActive: false, imageIndex: index}  
    );  
    this.displayImage(index);  
}
```

- ▶ This function is utilized when the User clicks an Image select point
- ▶ The Image displayed will be the one selected
- ▶ **displayImage (imageIndex)** is utilized to show the Image
- ▶ The carousel loop is cleared
  - ▶ Symptomatic of React behavior
  - ▶ Ensure the time does not decrease in a loop
  - ▶ The loop will restart with the previously enacted functionality in **componentDidMount()** starting with the current Image

# Six Functions (4 and 5)

User Moves Forward >

```
forwardImage() {  
    // User clicks the Forward Button  
  
    let index = 1;  
  
    this.setState({carouselActive: false});  
  
    if (this.state.imageIndex >=  
        this.state.numImages)  
    {  
        this.setState({imageIndex: index});  
    } else {  
        index = this.state.imageIndex + 1;  
        this.setState({imageIndex: index});  
    }  
  
    this.displayImage(index);  
}
```

About Movement

- These functions are utilized when the User clicks the Arrow Buttons
- Both functions utilize **displayImage(imageIndex)**
- The carousel loop is paused and restarted with the current Image as with **selectImage(Index)**

< User Moves Backward

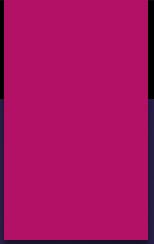
```
backwardImage() {  
    // User clicks the Backward Button  
  
    let index = this.state.imageIndex;  
  
    this.setState({carouselActive: false});  
  
    if (index <= 1) {  
        index = this.state.numImages;  
        this.setState({imageIndex:  
            this.state.numImages});  
    } else {  
        index = this.state.imageIndex - 1;  
        this.setState({imageIndex: index});  
    }  
  
    this.displayImage(index);  
}
```

# Six Functions (6)

```
displayImage(imageIndex) {  
  let images =  
    document.getElementsByClassName("carouselImages");  
  // All images hidden by default  
  for (let i = 0; i < images.length; i++) {  
    images[i].style.display = "none"; }  
  // Only show the current image  
  if (images.length > 0) {  
    images[imageIndex - 1].style.display = "block"; }  
  this.setState({carouselActive: false});  
}
```

- ▶ This function sets a specific Image to display
- ▶ Visibility
  - ▶ All Images are Hidden initially
  - ▶ Only the active Image is set to be Visible
- ▶ The carousel loop is paused and restarted with the current Image as with **selectImage()**

# Style



# Style

- ▶ Looking at specific styling utilized for the Transitions between the Images
- ▶ Using the CSS Keyframes

# Style

```
.imageTransition {  
    animation:  
        imageTransition  
        3s ease-in-out alternate;  
}  
  
@keyframes imageTransition {  
    0%, 100% { opacity: .8; }  
    40%, 60% { opacity: 1; }  
}
```

# Style

```
.imageTransition {  
    animation:  
        imageTransition  
        3s ease-in-out alternate;  
}  
  
@keyframes imageTransition {  
    0%, 100% { opacity: .8; }  
    40%, 60% { opacity: 1; }  
}
```

- ▶ The Class to be manipulated by the **keyframe** actions
- ▶ **animation** is the part the **keyframe** looks for
- ▶ In this Class
  - ▶ **animation-name** = **imageTransition** (name)
  - ▶ **animation-duration** = **3s** (length of animation)
  - ▶ **animation-timing-function** = **ease-in-out** (slow start and end)
  - ▶ **animation-direction** = **alternate** (forward then backwards)

# Style

```
.imageTransition {  
  animation:  
    imageTransition  
    3s ease-in-out alternate;  
}  
  
@keyframes imageTransition {  
  0%, 100% { opacity: .8; }  
  40%, 60% { opacity: 1; }  
}
```

- ▶ The **keyframe** actions
- ▶ Applies to the **imageTransition** animation
- ▶ During the animation time
  - ▶ Start at 0.8 opacity
  - ▶ Step to 1.0 opacity
  - ▶ Step back down to 0.8 opacity

# Setup

# Setup

```
import Carousel from  
  ".../components/Carousel";  
  
import {getCarouselImageNames} from  
  ".../utils/getConfig";
```

- ▶ To set the Carousel to appear on a Page like **Home.jsx**, import the necessary pieces
- ▶ Do this for every page wanted

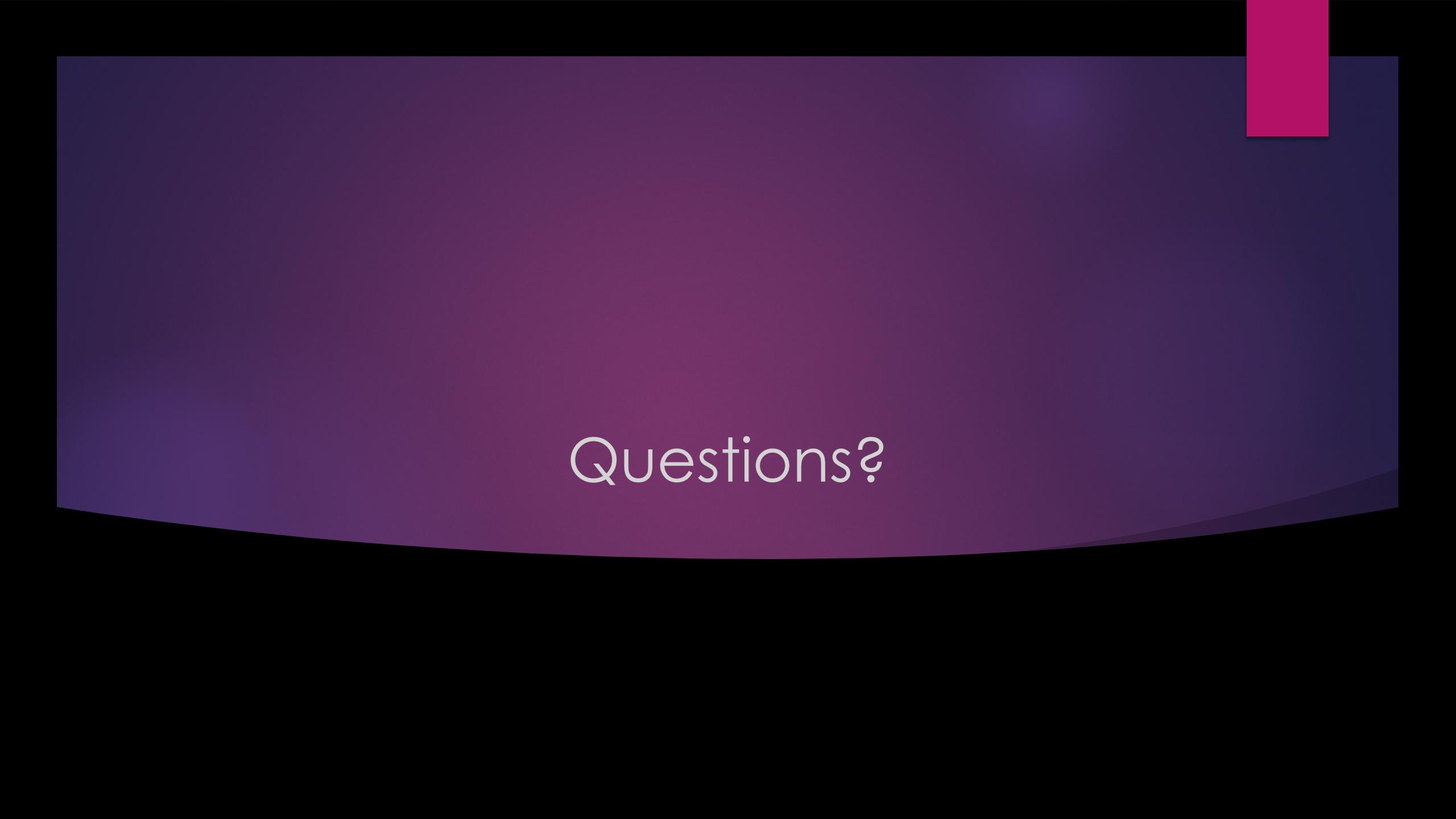
# Setup

```
let carousel= '';
if (getCarouselImageNames().length > 0) {
  carousel = <Carousel/>;
}
return (
  <div className={styles.contentcenter}>
    {carousel}
  </div>
);
```

- ▶ Check if there are Images set to show
- ▶ Show the Images if defined

# Example

[VIEW A FUNCTIONAL IMAGE CAROUSEL FOR GLTG](#)



Questions?

\*END\*