Zach Weber

Follow 11 Followers About

Creating A Rating Feature Using React.js and Material UI

Zach Weber Jun 26, 2020 · 3 min read ⋆

When I began building my app, MyParks Explorer, I knew that I would need to build a 5-star rating feature as part of my review component. I did a fair amount of research into how to build a rating component from scratch, but having already decided to use Material UI as my UI library, I decided to give their rating component a try.

I found Material's rating component to be exactly what I needed and easy to implement right out of the box, and how often can you say that? In addition to being easy to set up, Material also makes it quite simple to adjust and personalize the component. I'll walk you through the steps I took to set up the 3 different versions of my rating component.

Controlled rating for creating a review

When a user writes a review in my app, I wanted to give them the ability to rate a national or state park using a 5-star rating system. The component needs to register the value of the star that the user has clicked, then render that value. Out of the box, Material accomplishes this through React hooks: `const [value, setValue] = React.useState(0)`.

```
import React from 'react';
import Rating from '@material-ui/lab/Rating';
import Box from '@material-ui/core/Box';

export default function StarRating(props) {
  const [value, setValue] = React.useState(0);

  return (
    <div>
      <Box align="left"  component="fieldset" mb={3}
borderColor="transparent">
        <Rating
          value={value}
          name="rating"
          onChange={(event, newValue) => {
            setValue(newValue);
          }}
          onClick={props.handleInputChange}
        />
      </Box>
    </div>
  )
}
```

The `onChange` is what allows for the star that your mouse is hovered over to get highlighted. `onClick` handles the registering of the value of the star that you've selected. `value` displays the value of the star that you've selected.

Read-only rating for submitted reviews

Once a review has been submitted, the review (along with the user's rating) is visible on the reviewed park's page, as well as on the user's profile page. Since this rating shouldn't be able to be edited, it's set to read-only.

```
import React from 'react';
import Rating from '@material-ui/lab/Rating';
```

```
import Box from '@material-ui/core/Box';

export default function RenderStarRating(props) {
  return (
    <div>
      <Box align="left" mb={1} borderColor="transparent">
        <Rating
          value={props.reviewInfo.rating}
          name="rating"
          readOnly="true"
        />
      </Box>
    </div>
  )
}
```

As you can see, the `value` is set by accessing `props.reviewInfo.rating`. `readOnly` is set to `"true"`.

Creating and displaying an average user rating for a park

The functionality that took me the most time to put together was the overall rating of the park based on all submitted reviews for that particular park. To do this, I first created an algorithm that averages all ratings for that particular park:

```
getParkRating = () => {
  const sum = (accumulator, currentValue) => accumulator +
currentValue;
  const ratingsArr = this.props.showPark.reviews.map(reviewObj =>
reviewObj.rating)
    return ratingsArr.reduce(sum) / ratingsArr.length
}
```

Following that, I needed to set this average as the value of the rating component; however, with my app mostly being a side project and not having many users, I wanted to create a default value if the park hadn't been rated yet. To accomplish this I created a ternary statement that says if there is one or more reviews, set the rating to the average of the user ratings, otherwise, set the value to 4.5 stars: `value={this.props.showPark.reviews.length >= 1 ? this.getParkRating() : 4.5}`:

```
<Card>
  <CardContent>
    <Typography className={classes.bold} variant="h6">
      User Rating
    </Typography>
    <Divider />
    <Grid container>
      <Rating
        value={this.props.showPark.reviews.length >= 1 ?
this.getParkRating() : 4.5}
        name="rating"
        size="large"
        precision={0.5}
        readOnly="true"
      />
    </Grid>
  </CardContent>
</Card>
```

Similar to the previous rating component, `readOnly` is set to `"true"` because users shouldn't be able to alter the park's rating manually. Additionally, I altered the component by increasing the size to large (`size="large"`) and setting the precision to `{0.5}` so the average can show half-star ratings.

Conclusion

As you can see, Material UI makes implementing a rating component in your app rather simple. In most cases, the code can be utilized right out of the box, but if your needs differ, it's easy to make changes. Happy coding!

Reactjs Programming Ratings And Reviews Material Design


Download on the
App Store


GET IT ON
Google Play