How to pass data down

React is called React because each component ‘reacts’ to different data being passed into it.

Extracting FilmBrief from LandingPage

In LandingPage, we’re showing brief information on films. We should pull that JSX into its own component. But it needs the data that is actually resident in LandingPage. What should we do?

We should definitely pass each film into our new component through props.

1. The new JSX in LandingPage will look like this:

```
<FilmBrief film={film} key={film.id} />
```

2. Go ahead and extract your new FilmBrief component. Here’s a start:

```
export const FilmBrief = (props) => {}
```

3. That props will hold our film:

```
const film = props.film;
```

   Of course there a many ways to pull films from props. This is just the easiest way to understand.
4. Don’t forget to move the styles needed.
5. Run and test. It should look exactly as it did before but now it’s better designed.

Creating the ShowingTimes

Let’s do one that will be more challenging. Each movie has several showings per day. We should display all showings for each film for the currentDate.

6. Create a new component called ShowingTimes with all the required imports and exports and props and returned JSX and all that. The JSX should just be a placeholder for now.
7. Import ShowingTimes into FilmBrief.
8. Render a ShowingTimes into each FilmBrief.
9. Run and test. Make sure you can see a ShowingTimes for each FilmBrief.
10. We’ve put a ShowingTimes.jsx in the starters/html folder to save you some time. Copy the contents into your new ShowingTime.
11. Notice that the provided template needs an array called showingsForDateAndFilm. Put this above the returned JSX:

```
currentDate = new Date(currentDate);
const showingsForDateAndFilm = showings.filter(st => st.film_id ===
currentFilm.id && st.showing_time > currentDate.setHours(0, 0, 0, 0) &&
st.showing_time < currentDate.setHours(23, 59, 59, 999));
```

12. It also calls for some styles. Here you go!

```
wrapper: {
  padding: '5px',
},
headline: {
  margin: '0',
  fontWeight: 'bold',
  fontSize: '0.8em',
},
```

```
showingTimesWrapper: {
  display: 'flex',
  flexWrap: 'wrap',
},
tile: {
  background: 'rgba(0,0,0,0.25)',
  color: 'black',
  fontWeight: '200',
  fontSize: '0.8em',
  padding: '5px',
  margin: '3px',
},
```

13. Almost there. This component needs some props passed into it. Add them to the JSX. Make the call in FilmBrief look like this:

```
<div style={styles.showings}>
  {showings && <ShowingTimes showings={showings}
                             currentFilm={film} currentDate={currentDate} />}
</div>
```

If ShowingTimes needs them then FilmBrief needs them also and must get them through its props. The passing of props through several levels is very common with React.

14. This is how FilmBrief's JSX needs to look now. Make it so.

```
<FilmBrief film={film} showings={state.showings} currentDate={currentDate}
key={film.id} />
```

15. Once you think you're there, Run and test it. Make adjustments to get it to work.
16. Now, the big payoff! Reuse ShowingTimes in FilmDetail to show the ShowingTimes there also. (Hint: FilmDetails can know about the showings and the currentDate from state.)

Bonus! Decomposing Table and Seat

17. PickSeats is pretty complex. It would be much more understandable if we created a Table component. And then Table.js would be cleaner if we decomposed a Seat from it.
18. Create the Table.js and Seat.js components, use them in PickSeats, making sure to pass the props needed to each.
19. Don't forget to move any needed styles and any needed functions.