## Measure, improve, measure again, improve, measure once mo

```
\{data.map((dataPoint, index) \Rightarrow \{
 const className = classNames("HeatMap__DataPoint", {
   "HeatMap__DataPoint-isActive": activeDataPoint == index,
});
return (
   <div
     tabIndex="0"
     key={index}
     onMouseEnter=\{(evt) \Rightarrow handleInteraction(index, evt)\}
     onFocus=\{() \Rightarrow handleInteraction(index)\}
     onMouseLeave=\{() \Rightarrow handleInteraction(null)\}
     onBlur=\{() \Rightarrow handleInteraction(null)\}
     style={{
       width: blockWidth,
       backgroundColor: colorScale(dataPoint.number),
     className={className}
     aria-labelledby={tooltipId}
   ></div>
```

## Measure, improve, measure again, improve, measure once mo

```
\{data.map((dataPoint, index) \Rightarrow \{
const className = classNames("HeatMap__DataPoint", {
   "HeatMap__DataPoint-isActive": activeDataPoint === index,
return (
  <div
     tabIndex="0"
     key={index}
     onMouseEnter={(evt) ⇒ handleInteraction(index, evt)}
     onFocus={() ⇒ handleInteraction(index)}
     onMouseLeave=\{() \Rightarrow handleInteraction(null)\}
     onBlur=\{() \Rightarrow handleInteraction(null)\}
       width: blockWidth,
       backgroundColor: colorScale(dataPoint.number),
     className={className}
     aria-labelledby={tooltipId}
   ></div>
```

```
{data.map((dataPoint, index) \Rightarrow {
const isActive = state.activeDataPoint ≡ index;
return (
  <Cell
    key={`${dataPoint.time}-${dataPoint.rawValue}`}
     backgroundColor={colorScale(dataPoint.rawValue)}
     onMouseEnter={handleMouseEnter}
    index={index}
    onFocus={handleFocus}
     onBlur={resetTooltip}
    isActive={isActive}
    height={height}
    width={blockWidth}
    tooltipId={tooltipId}
```