```
handleWheel = (deltaX) => {
   const offset = (range.get(1) - range.get(0)) * 0.001 * (-deltaX)
   //处理时间轴滑动到最边缘的情况
   if (range.get(0).day() != moment(range.get(0) + offset).day() || range.get(1).day() != moment(range.get(1) +
offset).day()){return}
   this.setState({range: range.update(0, (v) => moment(v + offset)).update(1, (v) => moment(v + offset)) })
}
 getNotAllowPeriod = () => {
   const { week, periods, selectDate } = this.props
   if (!selectDate || !week) return fromJS([])
   let isOpenDay = week.get(WEEK[selectDate.format('d')])
  if (isOpenDay){
     let ONE_SECOND = 1000;let timeArr = [0]
     periods && periods.forEach((period) => {
       let startOfDay = moment(period.get('startTime')).startOf('day')
       timeArr.push(moment(period.get('startTime')).diff(startOfDay))
       timeArr.push(moment(period.get('endTime')).diff(startOfDay))})
     timeArr.push(24 * 60 * 60 * 1000 - ONE_SECOND); let startOfSelectDate = selectDate.startOf('day')
     for (let i = 0; i < timeArr.length; i += 2){
       if (timeArr[i] != timeArr[i + 1]){
         notAllow.push({startTime: moment(startOfSelectDate).add(timeArr[i], 'ms'),
           endTime: moment(startOfSelectDate).add(timeArr[i + 1], 'ms')
         })}}} else {notAllow.push({
         startTime: moment(selectDate).startOf('day'),
         endTime: moment(selectDate).endOf('day')})}
   return fromJS(notAllow)
 }
```