**BÁO CÁO TUẦN 3.2**

Tên nhóm: Nhóm 4

Thành viên

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Họ và tên | Email | SDT |
| 1 | Nguyễn Hữu Định | huudinh1401.tdc@gmail.com | 0961341401 |
| 2 | Nguyễn Văn Hải | nguyenhai1449@gmail.com | 0967678646 |
| 3 | Đặng Phương Đông | dpdong255@gmail.com | 0358846070 |

Email nhóm: ltdd3nhomhaidinh@gmail.com

Pass: nhomhaidinh

Github: ltdd3nhomhaidinh@gmail.com

Pass: ltdd3nhomhaidinh

**Tìm hiểu: Các Component cơ bản**

**Listview**

**a. Khái niệm**

React Native cung cấp một bộ component cho việc xuất ra danh sách các phần tử. Nói chung, bạn sẽ cần sử dụng “FlatList” hay “SectionList”.

Component “FlatList” xuất ra một danh sách cuộn có thể thay đổi tương tự với cấu trúc dữ liệu. “FlatList” làm việc tốt đối với những danh sách dữ liệu dài, nơi mà số hạng mục thay đổi theo thời gian. Không giống với “ScrollView” , “FlatList” chỉ trả về các phần tử đang hiển thị trên màn hình, không phải là tất cả các phần tử cùng lúc.

“FlatList” yêu cầu 2 thuộc tính: “data” và “renderItem”. “data” chứa các nguồn thông tin bên trong danh sách còn “renderItem” lấy một item trong đó và trả về một định dạng để hiển thị.

**b. Ví dụ**

class MyComponent extends Component {

constructor() {

super();

const ds = new ListView.DataSource({rowHasChanged: (r1, r2) => r1 !== r2});

this.state = {

dataSource: ds.cloneWithRows(['row 1', 'row 2']),

};

}

render() {

return (

<ListView

dataSource={this.state.dataSource}

renderRow={(rowData) => <Text>{rowData}</Text>}

/>

);

}

}

**c. Props**

* dataSource

Một phiên bản của ListView.DataSource để sử dụng

* initialListSize

Có bao nhiêu hàng để kết xuất trên mount thành phần ban đầu. Sử dụng công cụ này để làm cho nó có giá trị dữ liệu màn hình đầu tiên xuất hiện cùng một lúc thay vì qua nhiều khung.

* onEndReachedThreshold

Ngưỡng tính bằng pixel (ảo, không phải vật lý) để gọi onEndReached.

* pageSize

Số lượng hàng để kết xuất trên mỗi vòng lặp sự kiện. Lưu ý: nếu 'hàng' của bạn thực sự là các ô, tức là chúng không trải rộng toàn bộ chiều rộng của chế độ xem của bạn (như trong ListViewGridLayoutExample), bạn nên đặt kích thước trang là bội số của mỗi ô, nếu không bạn có thể thấy các khoảng trống ở rìa ListView khi các trang mới được tải.

* renderRow

(rowData, partID, rowID, highlightRow) => renderable

Lấy một mục nhập dữ liệu từ nguồn dữ liệu và id của nó và sẽ trả về một thành phần có thể kết xuất được hiển thị dưới dạng hàng. Theo mặc định, dữ liệu chính xác là những gì được đưa vào nguồn dữ liệu, nhưng cũng có thể cung cấp các trình trích xuất tùy chỉnh. ListView có thể được thông báo khi một hàng đang được tô sáng bằng cách gọi highlightRow (phầnID, rowID). Điều này đặt giá trị boolean của liền kềRowHighlighted trong renderSpayator, cho phép bạn điều khiển các dấu phân cách ở trên và dưới hàng được tô sáng. Trạng thái được tô sáng của một hàng có thể được đặt lại bằng cách gọi highlightRow (null).

* renderScrollComponent

Một hàm trả về thành phần có thể cuộn trong đó các hàng danh sách được hiển thị. Mặc định để trả về một ScrollView với các đạo cụ đã cho.

* scrollRenderAheadDistance

Làm thế nào sớm để bắt đầu kết xuất hàng trước khi chúng xuất hiện trên màn hình, tính bằng pixel.

* stickyHeaderIndices

Một loạt các chỉ số con xác định những đứa trẻ nào được neo vào đầu màn hình khi cuộn. Ví dụ: việc chuyển stickHeaderInices = {[0]} sẽ khiến đứa trẻ đầu tiên được cố định vào đầu chế độ xem cuộn. Thuộc tính này không được hỗ trợ kết hợp với horizonta = {true}.

* enableEmptySections

Cờ cho biết liệu tiêu đề phần trống sẽ được hiển thị. Trong bản phát hành tương lai, các tiêu đề phần trống sẽ được hiển thị theo mặc định và cờ sẽ không được chấp nhận. Nếu các phần trống không mong muốn được hiển thị, các chỉ mục của chúng sẽ được loại trừ khỏi đối tượng partID.

* onEndReached

Được gọi khi tất cả các hàng đã được hiển thị và danh sách đã được cuộn đến bên trong onEndReachedThrưỡng của phía dưới. Sự kiện cuộn bản địa được cung cấp.