**Height & Width**

Chiều rộng và dài của các thành phần sẽ xác định kích thước của các thành phần giao diện trên màn hình.

**Kích thước cố định**

Cách đơn giản nhất để chỉ định kích thước của các thành phần là sử dụng các tham số kích thước cố định width và height trong style. Tất cả kích thước được sử dụng trong React Native đều tuân theo đơn vị pixel khi hiển thị lên trên màn hình.

import React, { Component } from 'react';

import { AppRegistry, View } from 'react-native';

class FixedDimensionsBasics extends Component {

render() {

return (

<View>

<View style={{width: 50, height: 50, backgroundColor: 'powderblue'}} />

<View style={{width: 100, height: 100, backgroundColor: 'skyblue'}} />

<View style={{width: 150, height: 150, backgroundColor: 'steelblue'}} />

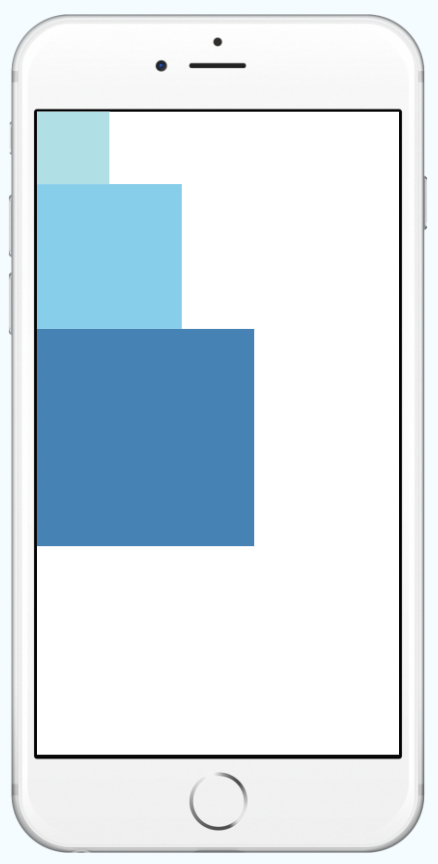
</View>

);

}

};

AppRegistry.registerComponent('AwesomeProject', () => FixedDimensionsBasics);



Để cài đặt việc hiển thị có sự tương tự nhau đối với nhiều loại kích thước màn hình các bạn đọc phần tiếp theo ngay dưới đây

**Kích thước linh động**

Sử dụng flex trong style của các thành phần để thanh đổi vùng hiển thị một cách linh động dựa trên vùng hiển thị có thể. Thông thường, bạn sẽ sử dụng flex: 1 khi mà bạn muốn một thành phần trải rộng hết mức có thể so với các thành phần khác có chung một view cha. Đây là giá trị tỷ lệ cao nhất mà flex cung cấp. Với các tỷ lệ khác thì nó sẽ so sánh với các thành phần ngang hàng với nó để đưa ra tỷ lệ hiển thị trong cùng một view cha.

Một thành phần chỉ có thể mở rộng ra hết khoảng trống có thể nếu như view cha của nó có kích thước lớn hơn 0. Nếu như view cha không được gán các kích thước cố định như width và height hoặc flex, thì kích thước của view cha sẽ có kích thước là 0 và khi sử dụng flex tại view con thì nó cũng sẽ không được hiển thị.

import React, { Component } from 'react';

import { AppRegistry, View } from 'react-native';

class FlexDimensionsBasics extends Component {

render() {

return (

// Try removing the `flex: 1` on the parent View.

// The parent will not have dimensions, so the children can't expand.

// What if you add `height: 300` instead of `flex: 1`?

<View style={{flex: 1}}>

<View style={{flex: 1, backgroundColor: 'powderblue'}} />

<View style={{flex: 2, backgroundColor: 'skyblue'}} />

<View style={{flex: 3, backgroundColor: 'steelblue'}} />

</View>

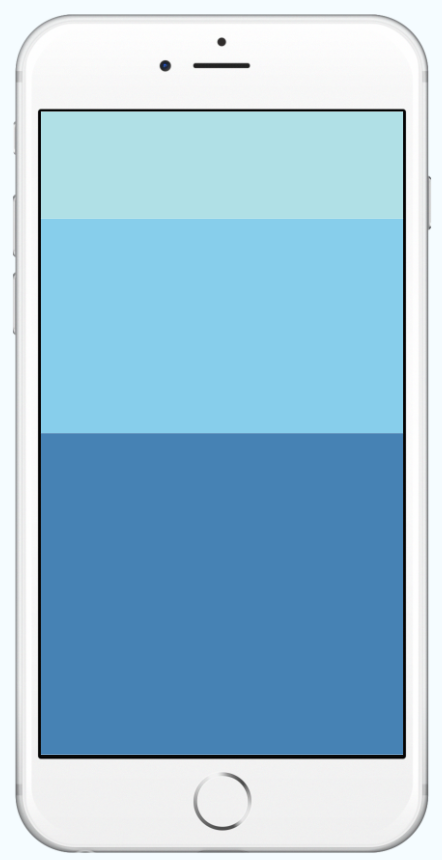
);

}

};

AppRegistry.registerComponent('AwesomeProject', () => FlexDimensionsBasics);

và kết quả chúng ta thu được ở hình dưới đây



Sau khi bạn đã có thể điều kiển được kích thước của các thành phần, bước tiếp theo chúng ta sẽ tìm hiểu cách làm thế nào để có thể thay đổi bố cục của các thành phần ở ngay phần dưới đây.

## Điều chỉnh Layout với Flexbox

Một thành phần có thể chỉ định rõ bố cục cho các view con của nó bằng cách sử dụng thuật toán của flexbox. Flexbox được thiết kế để cung cấp giao diện tương đồng trên những màn hình có kích thước khác nhau.

Bạn sẽ thường phải sử dung với các thuộc tính flexDirection, alignItems, và justifyContent để tạo ra được một giao diện đúng ý muốn.

Flexbox làm việc với React Native tương tự với cách mà CSS làm việc trên Web. nhưng cũng có một vài ngoại lệ. Giá trị mặc định là khác nhau, ví dụ như đối với thuộc tính flexDirection giá trị mặc định của nó sẽ là column thay thế cho giá trịnh row của CSS trên bản web, và giá trị của alignItems có giá trị mặc định là stretch chứ không phải là flex-start và tham số flexchỉ hỗ trợ số nguyên.

### Flex Direction

Thêm thuộc tính flexDirection vào style của một thành phần để thay đổi hướng sắp xếp các thành phần con của nó. Những thành phần con nên được sắp xếp trên cùng một hàng row hoặc hoặc trên cùng một cột column? Mặc định thì các thành phần con sẽ được sắp xếp nằm trên cùng một cột column.

import React, { Component } from 'react';

import { AppRegistry, View } from 'react-native';

class FlexDirectionBasics extends Component {

render() {

return (

// Try setting `flexDirection` to `column`.

<View style={{flex: 1, flexDirection: 'row'}}>

<View style={{width: 50, height: 50, backgroundColor: 'powderblue'}} />

<View style={{width: 50, height: 50, backgroundColor: 'skyblue'}} />

<View style={{width: 50, height: 50, backgroundColor: 'steelblue'}} />

</View>

);

}

};

AppRegistry.registerComponent('AwesomeProject', () => FlexDirectionBasics);

Chúng ta cùng xem kết quả thu được nhé

