REACT FLEXBOX

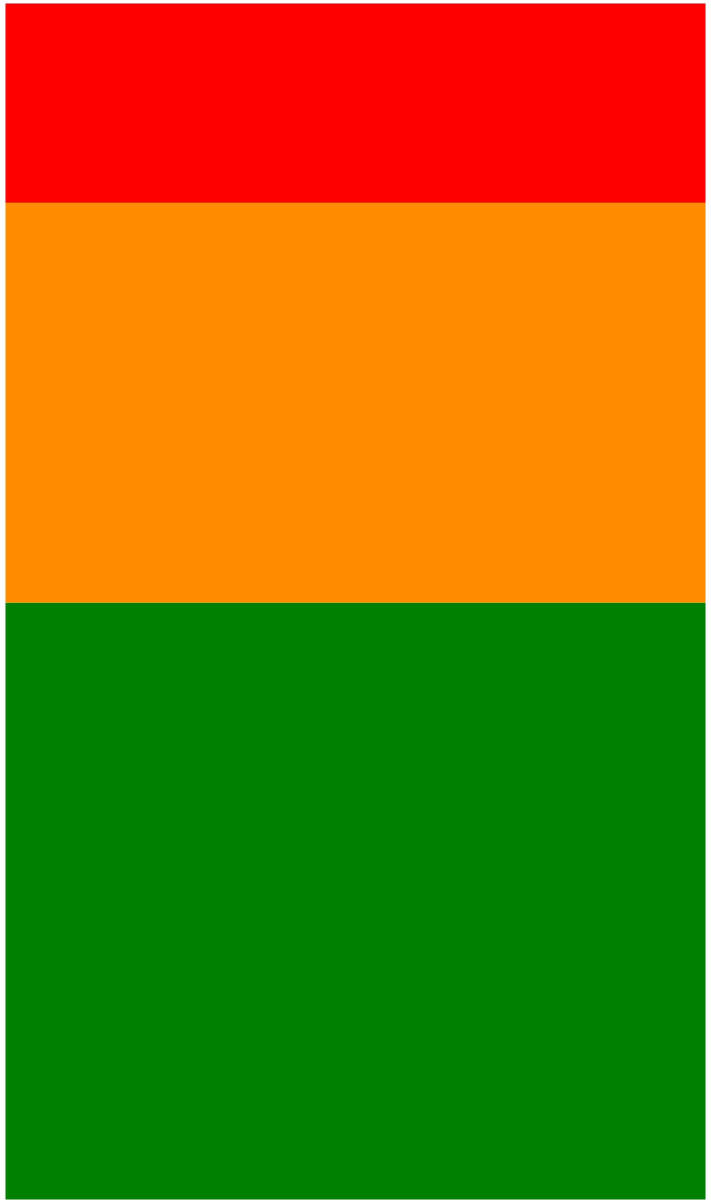
https://reactnative.dev/docs/flexbox

NGUYEN TRONG TIEN

```
import React from 'react';
import {StyleSheet, View} from 'react-native';

const Flex = () => {
  return (
    <View
      style={[
        styles.container,
        {
          // Try setting `flexDirection` to `"row"`.
          flexDirection: 'column',
        },
      ]}>
      <View style={{flex: 1, backgroundColor: 'red'}} />
      <View style={{flex: 2, backgroundColor: 'darkorange'}}
/>
      <View style={{flex: 3, backgroundColor: 'green'}} />
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 20,
  },
```



Flex Direction-flexDirection

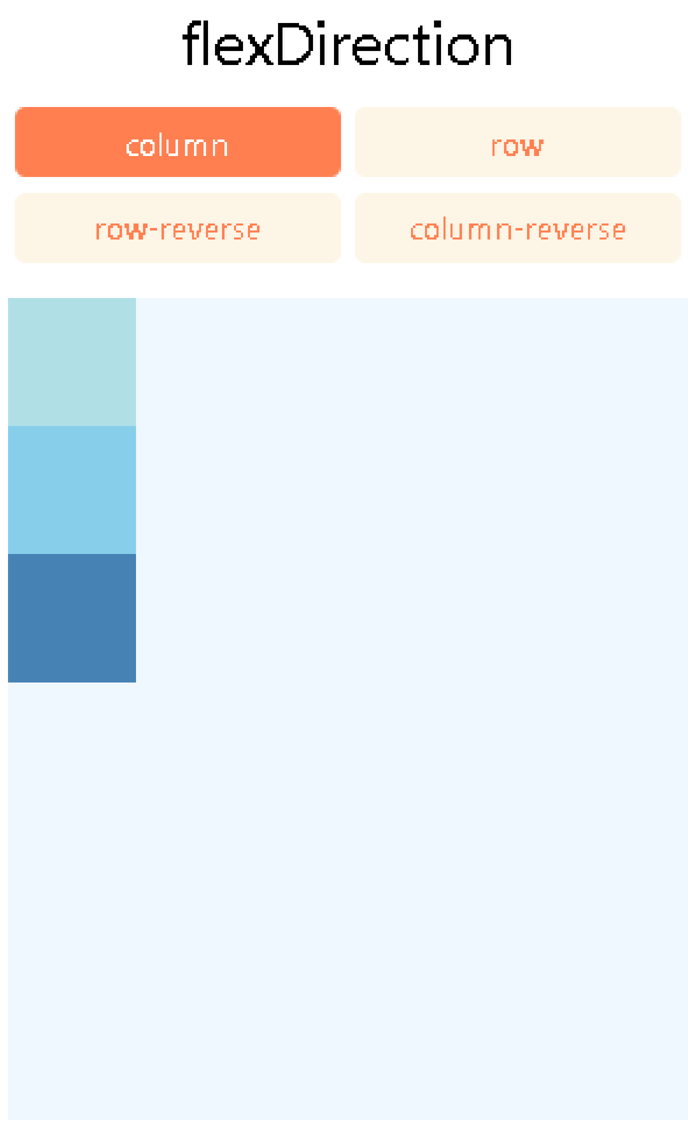
1. column (default value)
2. row Align children from left to right.
3. column-reverse
4. row-reverse

```
import React, {useState} from 'react';
import {StyleSheet, Text, TouchableOpacity, View} from 'react-native';
import type {PropsWithChildren} from 'react';

const FlexDirectionBasics = () => {
  const [flexDirection, setflexDirection] = useState('column');

  return (
    <PreviewLayout
      label="flexDirection"
      values={['column', 'row', 'row-reverse', 'column-reverse']}
      selectedValue={flexDirection}
      setSelectedValue={setflexDirection}>
      <View style={[styles.box, {backgroundColor: 'powderblue'}]} />
      <View style={[styles.box, {backgroundColor: 'skyblue'}]} />
      <View style={[styles.box, {backgroundColor: 'steelblue'}]} />
    </PreviewLayout>
  );
};

type PreviewLayoutProps = PropsWithChildren<{
  label: string;
  values: string[];
  selectedValue: string;
  setSelectedValue: (value: string) => void;
}>;
```



Justify Content-justifyContent

1. flex-start(default value)
2. flex-end
3. center
4. space-between
5. space-around
6. space-evenly

Align Items-alignItems

1. stretch (default value)
2. flex-start
3. flex-end
4. center
5. baseline

Align Self–alignSelf

alignSelf has the same options and effect as alignItems but instead of affecting the children within a container, you can apply this property to a single child to change its alignment within its parent. alignSelf overrides any option set by the parent with alignItems.

Align Content-alignContent

alignContent defines the distribution of lines along the cross-axis. This only has effect when items are wrapped to multiple lines using flexWrap.

1. flex-start (default value)
2. flex-end
3. stretch (default value when using Yoga on the web) S
4. center
5. space-between
6. space-around

Flex Wrap–flexWrap

1. The flexWrap property is set on containers and it controls what happens when children overflow the size of the container along the main axis. By default, children are forced into a single line (which can shrink elements). If wrapping is allowed, items are wrapped into multiple lines along the main axis if needed.
2. When wrapping lines, alignContent can be used to specify how the lines are placed in the container.

Flex Basis, Grow, and Shrink

1. flexBasis is an axis-independent way of providing the default size of an item along the main axis.
2. flexGrow describes how much space within a container should be distributed among its children along the main axis.
3. flexGrow accepts any floating point value >= 0, with 0 being the default value.
4. flexShrink describes how to shrink children along the main axis in the case in which the total size of the children overflows the size of the container on the main axis.
   flexShrink accepts any floating point value >= 0, with 0 being the default value (on the web, the default is 1). A container will shrink its children weighted by the children's flexShrink values.

Width and Height

- The width property specifies the width of an element's content area. Similarly, the height property specifies the height of an element's content area.

Both width and height can take the following values:

- auto (default value) React Native calculates the width/height for the element based on its content, whether that is other children, text, or an image.
- pixels Defines the width/height in absolute pixels. Depending on other styles set on the component, this may or may not be the final dimension of the node.
- percentage Defines the width or height in percentage of its parent's width or height, respectively.

Absolute & Relative Layout

The position type of an element defines how it is positioned within its parent.

- relative (default value) By default, an element is positioned relatively. This means an element is positioned according to the normal flow of the layout, and then offset relative to that position based on the values of top, right, bottom, and left. The offset does not affect the position of any sibling or parent elements.
- absolute When positioned absolutely, an element doesn't take part in the normal layout flow. It is instead laid out independent of its siblings. The position is determined based on the top, right, bottom, and left values.

THANK YOU!