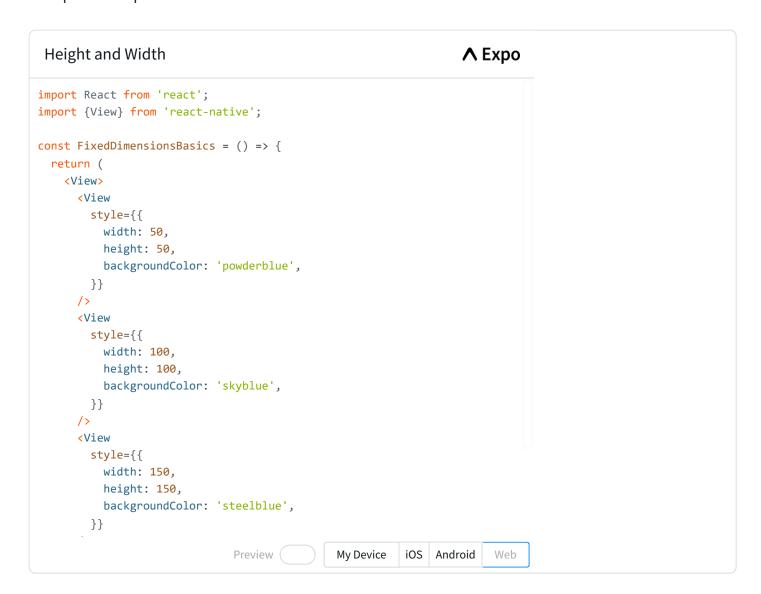# Height and Width

A component's height and width determine its size on the screen.

Fixed Dimensions

The general way to set the dimensions of a component is by adding a fixed `width` and `height` to style. All dimensions in React Native are unitless, and represent density-independent pixels.

Height and Width   **∧ Expo**

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

const FixedDimensionsBasics = () => {
  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',
        }}
```

Preview ⬤   My Device | iOS | Android | Web

Setting dimensions this way is common for components whose size should always be fixed to a number of points and not calculated based on screen size.

> ⚠️ **CAUTION**
>
> There is no universal mapping from points to physical units of measurement. This means that a component with fixed dimensions might not have the same physical size, across different devices and screen sizes. However, this difference is unnoticeable for most use cases.

Flex Dimensions

Use `flex` in a component's style to have the component expand and shrink dynamically based on available space. Normally you will use `flex: 1`, which tells a component to fill all available space, shared evenly amongst other components with the same parent. The larger the `flex` given, the higher the ratio of space a component will take compared to its siblings.

> ⓘ **INFO**
>
> A component can only expand to fill available space if its parent has dimensions greater than `0`. If a parent does not have either a fixed `width` and `height` or `flex`, the parent will have dimensions of `0` and the `flex` children will not be visible.

Flex Dimensions   ∧ Expo

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

const FlexDimensionsBasics = () => {
  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>
  );
};

export default FlexDimensionsBasics;
```

Preview ⬭   My Device | iOS | Android | Web

After you can control a component's size, the next step is to learn how to lay it out on the screen.

Percentage Dimensions

If you want to fill a certain portion of the screen, but you *don't* want to use the `flex` layout, you *can* use **percentage values** in the component's style. Similar to flex dimensions, percentage dimensions require parent with a defined size.

Percentage Dimensions

∧ Expo

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

const PercentageDimensionsBasics = () => {
  // Try removing the `height: '100%'` on the parent View.
  // The parent will not have dimensions, so the children can't expand.
  return (
    <View style={{height: '100%'}}>
      <View
        style={{
          height: '15%',
          backgroundColor: 'powderblue',
        }}
      />
      <View
        style={{
          width: '66%',
          height: '35%',
          backgroundColor: 'skyblue',
        }}
      />
      <View
        style={{
          width: '33%',
          height: '50%',
          backgroundColor: 'steelblue',
```

Preview ⬤ | My Device | iOS | Android | Web

Is this page useful? 👍 👎

✏️ Edit this page

*Last updated on **Jun 21, 2023***