



React Native Styling

Silver- Chapter 2 - Topic 5

**Selamat datang di Chapter 2 Topic 5 pada course
React Native dari Binar Academy!**





Haalloo teman-teman! 🙌

Di Topic 4 kita sudah bahas tentang fundamental React Native. Di Chapter 2 Topic 5 ini, kita bakal lanjut ke materi tentang **React Native Styling** yang penting banget dalam proses pembuatan layout biar aplikasimu kece. Penasaran?

Oke yaudah let's start!



Detailnya, kita bakal bahas hal-hal berikut ini:

- Beda Flexbox pada CSS dan React Native
- Cara kerja React Navigation
- React Native UI Framework





Untuk bisa bikin layout pada aplikasi, kamu harus tahu yang namanya **Layout Flexbox** dan paham bagaimana pengaplikasiannya. Yuklah, kita mulai pelajaran hari ini!

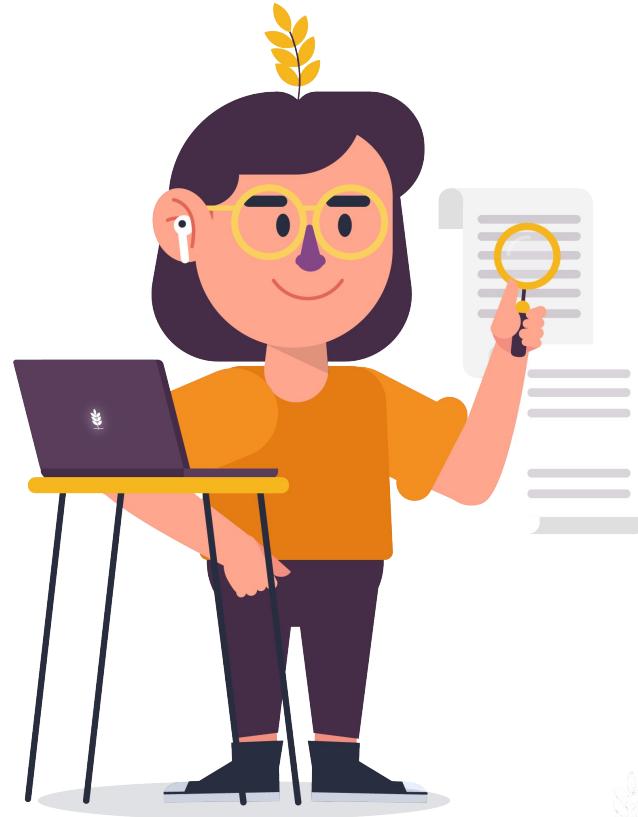




Walaupun sama-sama flexbox, tapi penggunaannya pada CSS dan pada React Native ada bedanya lho!

kalau di CSS kan flexbox digunakan untuk bikin layout berdasarkan column dan row. Nah di React Native, flexbox merupakan konsep utama dan satu-satunya untuk bikin layout.

Meski keduanya butuh flexbox, tapi ada sedikit perbedaan yang perlu kamu tahu antara penggunaan flexbox pada CSS dan pada React Native nih. Apa saja?





Kita cari tahu pengertian Flexbox dulu deh..

Flexbox adalah metode default (udah dari sananya) pada react native, sehingga kamu nggak perlu lagi untuk nambahin display: **flex** ketika melakukan styling.

```
const styles = StyleSheet.create({
  card: {
    display: 'flex' // ini tidak perlu dituliskan, karena pada react native semua display value nya udah 'flex'
  }
});
```



Ini lho bedanya Flexbox CSS vs Flexbox React Native

Property	Default Value di CSS	Default Value di React Native
flexDirection	Row	Column
alignContent	Stretch	Flex-start
flexShrink	1	0

Keterangan:

Pada CSS **flexDirection**, kalau nggak ditulis styling-nya, maka secara otomatis dia **akan menggunakan 'row'** sebagai default direction. Pada react native, kalau flexDirection nggak ditulis maka react native **akan menggunakan 'column'** sebagai default direction. Begitu juga untuk keterangan pada perbedaan alignContent dan flexShrink.



Kita bedah Flex Property pada React Native dulu ya~

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

export default function App() {
  return (
    <>
      <View style={{ backgroundColor: "#7cb48f", flex: 1 }} />
    </>
  );
}
```

Property **flex** pada **<view>** menentukan ukuran dari view pada layar device.

Contoh, kode di atas akan **menampilkan view warna hijau yang memenuhi layar device**, kok bisa? Karena view tersebut **punya property flex: 1**, artinya view akan mengambil semua space yang tersedia di layar device.





Ini adalah contoh lainnya~

```
export default function App() {
  return (
    <>
      <View style={{ backgroundColor: "#7cb48f", flex: 1 }} />
      <View style={{ backgroundColor: "#7CA1B4", flex: 3 }} />
    </>
  );
}
```



Kalau kita lihat kode di atas, **ada dua <view> yang punya nilai flex yang berbeda.** Flex: 3 yang terdapat pada View di baris kedua punya ukuran 3 kali lebih besar dibandingan View dengan flex: 1.

Coba perhatikan gambar di samping deh, warna biru jadi 3 kali lebih besar daripada warna hijau!



Sekarang saatnya latihan! Kita bikin layout sederhana yukk

Buatlah sebuah Box Card seperti gambar di samping. Sebelum mulai, pastikan kamu sudah download project React Native baru ya. Sebagai arahan, ikuti langkah di slide-slide berikutnya ya.





1. Buka file **app.js**, kemudian ubah kode yang berada di dalam app.js mengikuti struktur disamping ini ya.

URL Image :

<https://images.pexels.com/photos/3225517/pexels-photo-3225517.jpeg?cs=srgb&dl=pexels-michael-block-3225517.jpg>

```
// App.js

import React from "react";
import { StyleSheet, View, Text, Image, Dimensions } from "react-native";

export default function App() {
  return (
    <View style={styles.container}>
      <View style={styles.card}>
        <View>
          <Text style={styles.nameText}>React Native School</Text>
          <Text style={styles.followText}>Follow</Text>
        </View>
        <Image
          style={styles.image}
          resizeMode="cover"
          source={{
            uri:"https://images.pexels.com/photos/3225517/pexels-photo-3225517.jpeg?cs=srgb&dl=pexels-michael-block-3225517.jpg",
          }}
        />
        <View>
          <Text>
            <Text style={styles.nameText}>React Native School</Text>
            This has been a tutorial on how to build a layout with Flexbox. I
            hope you enjoyed it!
          </Text>
        </View>
      </View>
    </View>
  );
}
```



2. Masih di file app.js, **tulis basic styling** untuk bikin layout dasar. Kamu bisa ikutin kode di bawah ini ya.

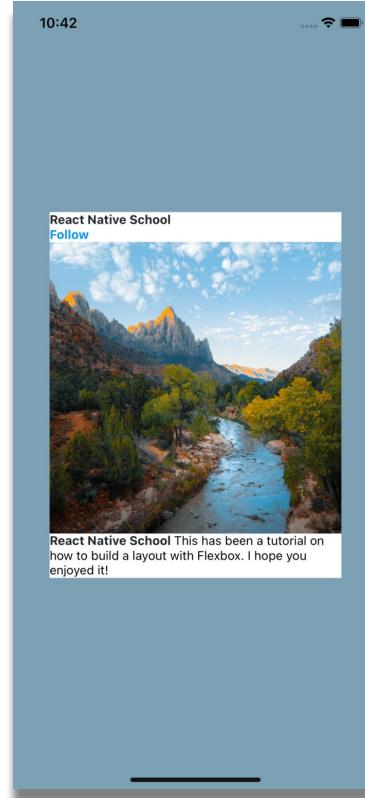
```
// App.js

// Tulis kode styling berikut dibawah component App

const screen = Dimensions.get("screen");
const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: "#7CA1B4",
    alignItems: "center",
    justifyContent: "center",
  },
  card: {
    backgroundColor: "#fff",
    width: screen.width * 0.8,
  },
  image: {
    height: screen.width * 0.8,
  },
  nameText: {
    fontWeight: "bold",
    color: "#20232a",
  },
  followText: {
    fontWeight: "bold",
    color: "#0095f6",
  },
});
```



3. Nah, setelah kamu ikuti dua langkah di atas, maka tampilan layout mobile-mu akan persis seperti gambar di samping.





4. Jangan lupa **atur text “Follow”** yang ada pada header ya, kamu bisa buat sejajar dengan text “React Native School”.





5. **Edit component App.** Tambahkan `styles.header` pada `<View>` yang membungkus kedua text “React Native Schoold” dan “Follow”

```
// App.js

import React from "react";
import { StyleSheet, View, Text, Image, Dimensions } from "react-native";

export default function App() {
  return (
    <View style={styles.container}>
      <View style={styles.card}>
        <View style={styles.header}>
          <Text style={styles.nameText}>React Native School</Text>
          <Text style={styles.followText}>Follow</Text>
        </View>
        <Image
          style={styles.image}
          resizeMode="cover"
          source={{
            uri:"https://images.pexels.com/photos/3225517/pexels-photo-3225517.jpeg?cs=srgb&dl=pexels-michael-block-3225517.jpg",
          }}
        />
        <View>
          <Text>
            <Text style={styles.nameText}>React Native School</Text>
            This has been a tutorial on how to build a layout with Flexbox. I
            hope you enjoyed it!
          </Text>
        </View>
      </View>
    </View>
  );
}
```



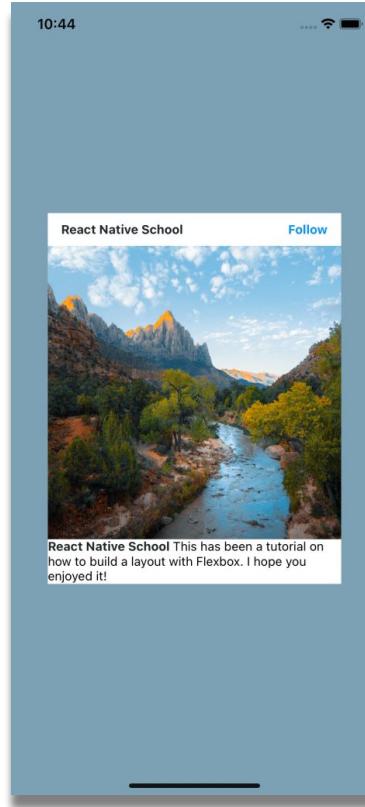
6. Pada bagian styling, **tambahkan property object style "header"**, lalu kamu bisa gunakan **flexDirection** dan **justifyContent** untuk bikin text “React Native School” dan “Follow” menjadi sejajar.

```
// App.js

const screen = Dimensions.get("screen");
const styles = StyleSheet.create({
    // ... kode styling kamu sebelumnya
    // tambahkan properti object style header dibagian
    // paling bawah kode styling kamu
    header: {
        flexDirection: "row",
        justifyContent: "space-between",
        paddingHorizontal: 15,
        paddingVertical: 10,
    },
});
```



- Setelah mengikuti langkah ke-6, maka tampilan layout aplikasi kamu akan terlihat kayak gambar di samping nih. Text “React Native School” dan “Follow” pada **bagian header berhasil jadi sejajar.**





8. Selanjutnya kamu tinggal **tambahin padding pada text** yang ada di bawah gambar.

Kamu akan bikin sebuah property object style baru dengan nama **footer**. Tambahkan property footer pada `<View>` yang membungkus kedua element `<Text>` yang ada di bawah `<Image>`.

```
// App.js

import React from "react";
import { StyleSheet, View, Text, Image, Dimensions } from "react-native";

export default function App() {
  return (
    <View style={styles.container}>
      <View style={styles.card}>
        <View style={styles.header}>
          <Text style={styles.nameText}>React Native School</Text>
          <Text style={styles.followText}>Follow</Text>
        </View>
        <Image
          style={styles.image}
          resizeMode="cover"
          source={{
            uri:"https://images.pexels.com/photos/3225517/pexels-photo-3225517.jpeg?cs=srgb&dl=pexels-michael-block-3225517.jpg",
          }}
        />
        <View style={styles.footer}>
          <Text>
            <Text style={styles.nameText}>React Native School</Text>
            This has been a tutorial on how to build a layout with Flexbox. I
            hope you enjoyed it!
          </Text>
        </View>
      </View>
    </View>
  );
}
```



9. Terakhir, pada bagian styling, **tambahkan property object style “footer”**, lalu tambahkan `paddingHorizontal` dan `paddingVertical` di dalamnya.

```
// App.js

const screen = Dimensions.get("screen");
const styles = StyleSheet.create({
    // ... kode styling kamu sebelumnya
    header: {
        flexDirection: "row",
        justifyContent: "space-between",
        paddingHorizontal: 15,
        paddingVertical: 10,
    },
    // tambahkan properti object style footer dibagian
    // paling bawah kode styling kamu
    footer: {
        paddingHorizontal: 15,
        paddingVertical: 10,
    }
});
```



This is it!

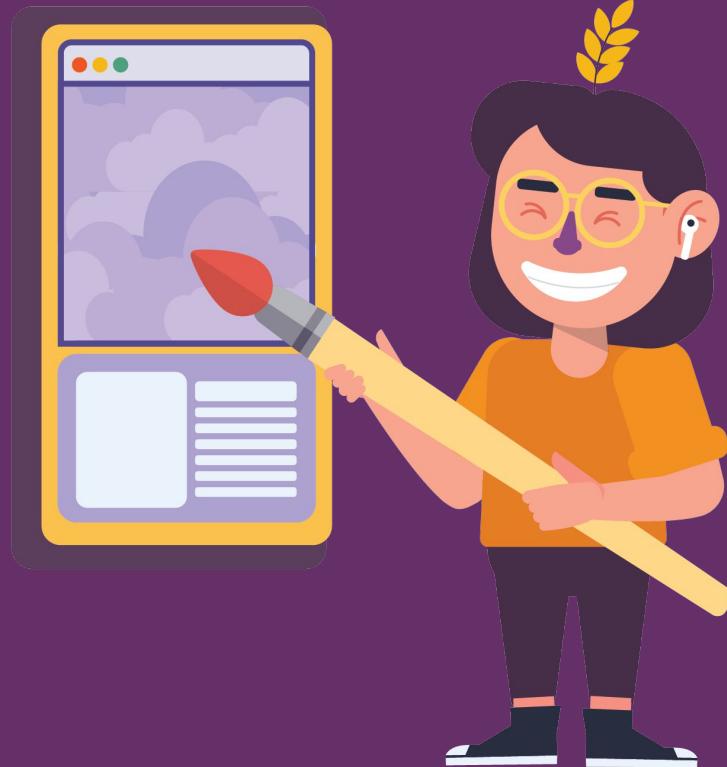
Sekarang layout card yang kamu buat sudah jadi deh!

Tampilannya bakal persis seperti gambar di samping ini.





Setelah layout, ada yang nggak kalah penting nih, yaitu **pembuatan Icons** supaya aplikasimu mudah ditemukan, diakses, dan dipakai oleh pengguna.





Ternyata selama ini kita dekat banget sama Icons ini, guys!

Yash, pasti hampir setiap hari kamu buka aplikasi, minimal aplikasi ojek online atau e-commerce, kan? Berangkat ke kampus atau kantor lewat aplikasi. Jajan makanan yang lagi hits di Tiktok pakai aplikasi. Pokoknya semua yang online pasti akan butuh aplikasi.

Nah untuk menjalankan aplikasi itulah, maka kamu perlu klik elemen gambar-gambar yang disebut dengan icons.





Mari kenalan sama Icons~

Icons adalah tampilan berupa **simbol atau gambar yang melambangkan fungsi sebuah objek**, dalam hal ini objeknya adalah aplikasi. Icons juga befungsi untuk memudahkanmu mengakses atau memberi perintah.

Karena itulah, Icon jadi salah satu elemen yang sangat penting dalam aplikasi. Icon biasanya terletak pada menu sebelah bawah atau sebelah samping.





Lalu gimana ya caranya nambahin Icon pada aplikasi React Native?

Begini..



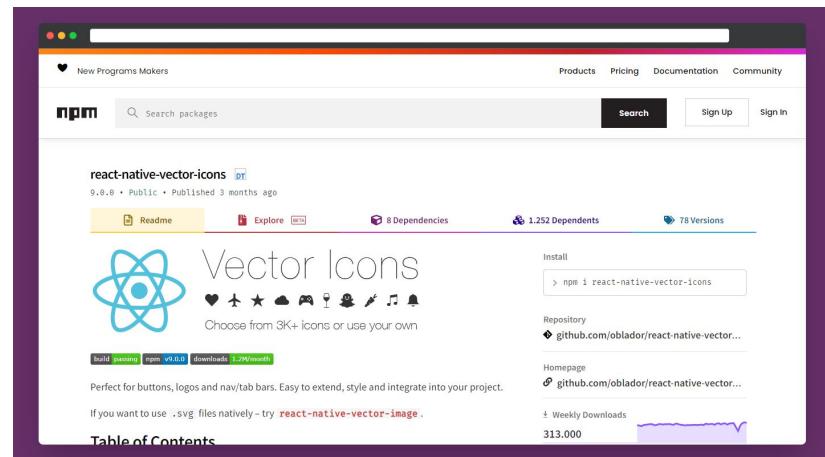


Icons di react native ada di dalam sebuah library, namanya React Native Vector Icons!

Di React Native, kamu bisa pakai package/library yang namanya **react-native-vector-icons**.

Package ini juga menyediakan banyak banget variasi icons yang sudah umum dipakai, seperti Material Icons, Ion Icons, Fontawesome, dll.

Biar makin kebayang, coba sekarang kamu install react-native-vector-icons pada aplikasimu yuk!





Jangan sampe lupa~

Sebelum kamu menginstall sebuah package/library, pastikan kamu memeriksa dokumentasi resmi dari package yang akan diinstall ya, karena cara **instalasi dan konfigurasi yang benar akan selalu ada di dalam dokumentasi resmi**.

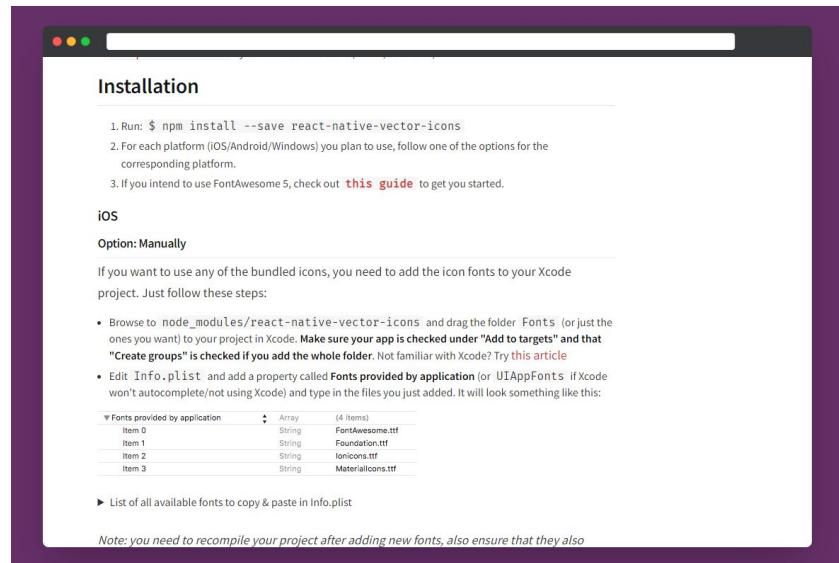




Malu bertanya, sesat di jalan Malas membaca, sesat installan

Guys, pastiin juga kamu baca dokumentasi untuk menginstall package react-native-vector-icons ya.

Kamu bisa lihat dokumentasinya pada website resmi npm atau bisa melalui link berikut:
<https://www.npmjs.com/package/react-native-vector-icons>



The screenshot shows a web page with a dark purple header and a white content area. The title 'Installation' is at the top. Below it is a numbered list of steps:

1. Run: \$ npm install --save react-native-vector-icons
2. For each platform (iOS/Android/Windows) you plan to use, follow one of the options for the corresponding platform.
3. If you intend to use FontAwesome 5, check out [this guide](#) to get you started.

Below the list, there's a section for 'iOS' with the heading 'Option: Manually'. It instructs users to add icon fonts to their Xcode project. A bulleted list provides specific steps for Xcode:

- Browse to `node_modules/react-native-vector-icons` and drag the folder `Fonts` (or just the ones you want) to your project in Xcode. Make sure your app is checked under "Add to targets" and that "Create groups" is checked if you add the whole folder. Not familiar with Xcode? Try [this article](#)
- Edit `Info.plist` and add a property called `Fonts provided by application` (or `UIAppFonts` if Xcode won't autocomplete/not using Xcode) and type in the files you just added. It will look something like this:

Fonts provided by application	Array	(4 items)
Item 0	String	FontAwesome.ttf
Item 1	String	Foundation.ttf
Item 2	String	Ionicons.ttf
Item 3	String	MaterialIcons.ttf

Below the table, a note says: ▶ List of all available fonts to copy & paste in Info.plist

Note: you need to recompile your project after adding new fonts, also ensure that they also



Yuk, langsung install~

Berikut adalah cara **instalasi React Native Vector Icons**.

Cus, langsung sambil praktekin, guys!

1. Buka terminal/cmd, kemudian install library React Native Vector Icons pada project.
2. Setelah install, lakukan linking package react-native-vector-icons.

```
$npm i react-native-vector-icons  
// atau  
$yarn add react-native-vector-icons
```

```
// ketikan perintah berikut pada terminal/cmd  
  
$npx react-native link react-native-vector-icons
```



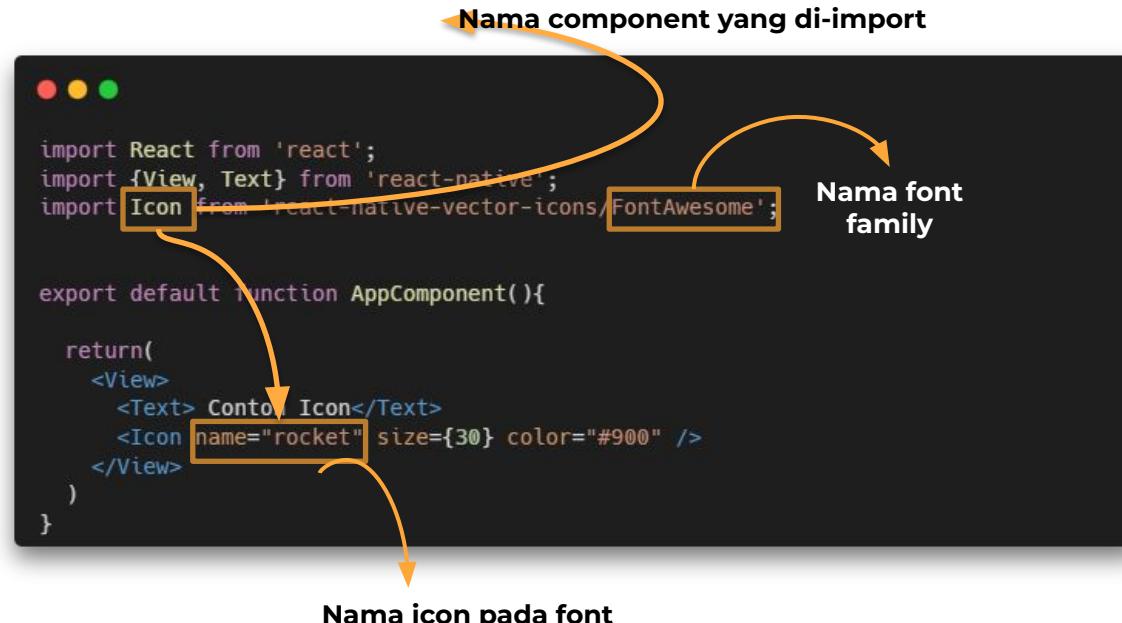
3. Kemudian build ulang project kamu.

```
● ● ●  
//build ulang android  
$npm run android
```

```
● ● ●  
//build ulang ios  
$npm run ios
```



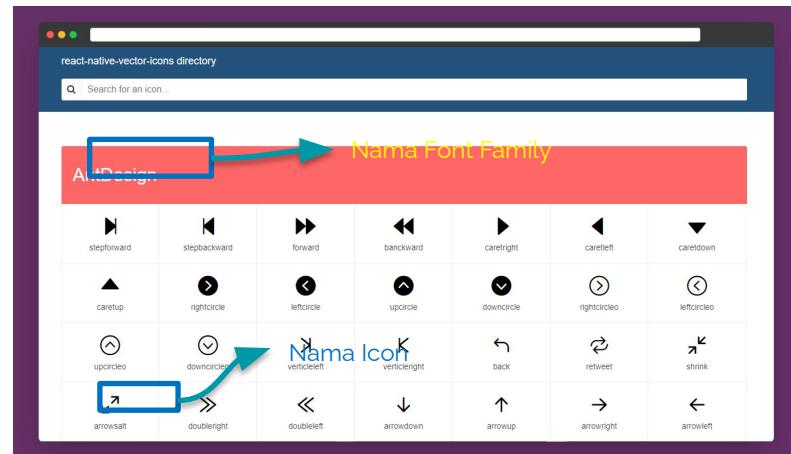
- Setelah instalasi selesai, kamu bisa menggunakan icon dengan cara memanggil icon tersebut.





- Untuk melihat semua icon yang ada pada react native icon, kamu bisa buka website ini ya:

<https://oblador.github.io/react-native-vector-icons/>





Gimana? Berhasil nggak bikin icons dengan langkah di atas?

Kalau sudah berhasil, sekarang kita pindah ke materi belajar selanjutnya, **Navigation**.





Ada 2 hal utama di Navigation~

Di materi navigations ini, kita akan bahas tentang dua hal utama dulu ya, yaitu:

- Navigate Screen di React Native
- Cara install React Navigation.





Kalau pindah rumah caranya pakai mobil box, Kalau pindah halaman aplikasi caranya pakai navigasi~

Navigasi screen adalah **cara agar aplikasi kamu bisa pindah-pindah halaman** ketika user mengklik tombol atau menu tertentu.

Untuk mengimplementasi navigasi screen pada React Native, sebenarnya kamu bisa pilih antara 2 package/library yang tersedia, yaitu: **react-navigation** atau **react-native-navigation**

tapi di sesi kali ini kita akan bahas tentang react-navigation aja ya!





Hal pertama yang harus kamu lakukan adalah **instal React Navigation**, kamu bisa ikuti langkah-langkah berikut ini ya!





1. Buka terminal/cmd, lalu install library React Native Vector Icons pada project kamu.



```
npm install @react-navigation/native  
// atau apabila kamu menggunakan yarn  
yarn add @react-navigation/native
```



2. Install juga dependency package/library lain yang dibutuhkan, yaitu react-native-screens dan react-native-safe-area-context.



```
$ npm install react-native-screens react-native-safe-area-context
// atau apabila kamu menggunakan yarn
$ yarn add react-native-screens react-native-safe-area-context
```

3. Kalau kamu juga mau develop aplikasi react native untuk platform iOS, maka kamu harus install dependency pods, dengan cara mengetik perintah npx pod-install ios.



```
$ npx pod-install ios
```



4. react-native-screens yang sudah kamu install sebelumnya, butuh tambahan konfigurasi Java pada Android.

Edit file **MainActivity.java** ini bisa kamu temuin pada folder:

android/app/src/main/java/<nama package android kmau>/MainActivity.java

Setelah itu build ulang aplikasi kamu.

```
1 package com.aldipee.supertodoapp;
2
3 import android.os.Bundle; // Tambahkan import ini paling pada statement import paling atas
4 import com.facebook.react.ReactActivity;
5
6 public class MainActivity extends ReactActivity {
7
8     // Tambahkan code ini didalam Class MainActivity
9     @Override
10    protected void onCreate(Bundle savedInstanceState) {
11        super.onCreate(null);
12    }
13
14 /**
15 * Returns the name of the main component registered from JavaScript. This is used to schedule
16 * rendering of the component.
17 */
18 @Override
19 protected String getMainComponentName() {
20     return "SuperTodoApp";
21 }
22 } You, 2 days ago • initit: initialize commit
```



- Setelah proses instalasi selesai, modifikasi kode pada file App.js atau file root project-mu, jadikan <NavigationContainer> sebagai bungkus utama component-component lain yang ada pada file tersebut.

```
// App.js atau root file project kamu

import React from 'react';
import { NavigationContainer } from '@react-navigation/native';

export default function App() {
  return (
    <NavigationContainer>{/* kode-kode componentn */}</NavigationContainer>
  );
}
```



Ngomongin transisi, tahu nggak sih? Kamu bisa lho bikin transisi pada layar dengan Stack Navigator.

Eh, stack navigator? Apa dan bagaimana tuh cara bikinnya?



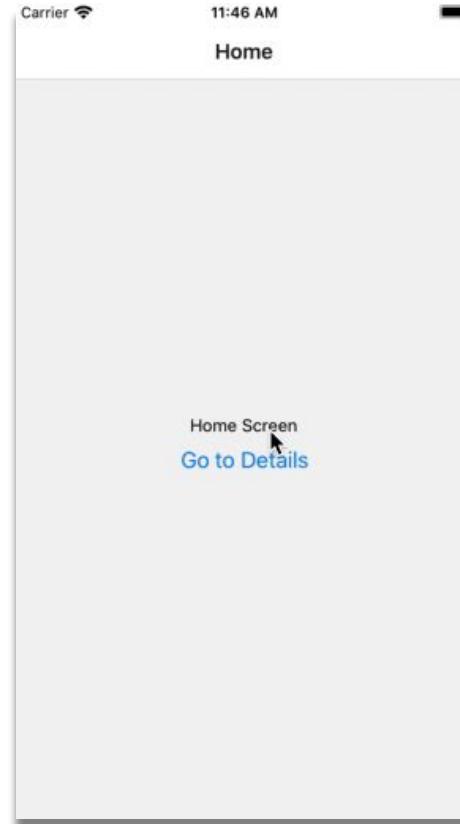


Abang ‘Stack Navigator’, antar aku ke screen sebelah ya~

Saat pakai aplikasi, nggak mungkin dong kamu ada di halaman yang itu-itu saja. Pasti kamu pengen pindah ke halaman lain supaya bisa mengakses informasi lebih banyak tentang suatu hal.

Nah, ini nih fungsi stack navigator.

Stack Navigator menyediakan **cara supaya aplikasimu bisa bertransisi** alias pindah-pindah antar screen. Setiap screen baru bakal ditempatkan di atas Stack.





Jangan lupa, sebelum bikin Stack Screen, kamu harus **install lebih dulu package tambahan react-navigation** yang berfungsi untuk membuat Stack ya.

```
$ npm install @react-navigation/native-stack
```



Kalau sudah terinstall, sekarang kamu bisa mulai **bikin screen pada React navigation**.

Untuk bikin screen, kamu harus membuat sebuah component yang akan ditampilkan sebagai satu halaman penuh. **Kode di bawah ini adalah component yang nantinya akan ditampilkan pada screen 'Home'.**

```
function HomeScreen() {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Home Screen</Text>
    </View>
  );
}
```



Kemudian, kamu **import createNativeStackNavigator, dan simpan pada sebuah variabel**, beri nama variabel tersebut Stack. Perlu dicatat pemanggilan function createNativeStackNavigator() **harus di luar dari sebuah component**, component apapun itu.

```
// Import ini createNativeStackNavigator
import { createNativeStackNavigator } from '@react-navigation/native-stack';

const Stack = createNativeStackNavigator();
```



Nah, sekarang kamu bisa pakai variable stack yang sudah kamu buat sebagai sebuah component yang berada di dalam `<NavigationContainer>`.

Perlu diingat ya, **setiap screen yang kamu buat harus memiliki nama unik masing-masing.** Nggak boleh ada yang sama, nanti codingan kamu bisa error

```
function HomeScreen() {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Home Screen</Text>
    </View>
  );
}

const Stack = createNativeStackNavigator();
```

```
function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator>
        <Stack.Screen name="Home" component={HomeScreen} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}

export default App;
```

Nama Screen yang kamu definisikan



Kayak gini nih kode akhir dan hasil tampilannya pada mobile apps kamu

```
// App.js atau root file project kamu

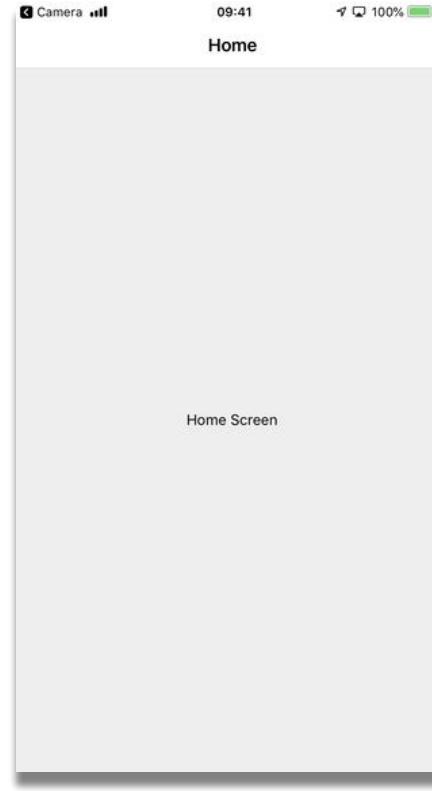
import * as React from 'react';
import { View, Text } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
// Import int createNativeStackNavigator
import { createStackNavigator } from '@react-navigation/native-stack';

function HomeScreen() {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Home Screen</Text>
    </View>
  );
}

const Stack = createStackNavigator();

function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator>
        <Stack.Screen name="Home" component={HomeScreen} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}

export default App;
```





Eh tapi..

**Itu kan baru satu screen ya, terus kalau mau
bikin lebih dari satu screen gimana dong?
Nanti pindah-pindah screen-nya gimana?**

Mari kita bahas..





Menambahkan Screen lain

Kamu nggak perlu bingung, guys! Sebab **nggak ada batasan dalam jumlah pembuatan screen kok**. Jadi pada dasarnya kamu bisa nambahin berapapun screen yang kamu perlu.

Caranya, kamu cuma perlu **bikin <Stack.Screen> baru, kemudian letakan <Stack.Screen> baru tersebut sejajar dengan <Stack.Screen> lainnya**.

```
<Stack.Screen name="NamaScreenKamuBebas" component={NamaComponentScreenKamu} />
```

```
<NavigationContainer>
  <Stack.Navigator initialRouteName="ScreenTiga">
    <Stack.Screen name="ScreenSatu" component={ComponentBuatScreenSatu} />
    <Stack.Screen name="ScreenDua" component={ComponentBuatScreenDua} />
    <Stack.Screen name="ScreenTiga" component={ComponentBuatScreenTiga} />
    <Stack.Screen name="ScreenEmpat" component={ComponentBuatScreenEmpat} />
    <Stack.Screen name="ScreenLima" component={ComponentBuatScreenLima} />
  </Stack.Navigator>
</NavigationContainer>
```



Pindah-pindah Screen

Selain menambahkan, kamu juga bisa memindahkan screen pada React Native. Untuk ini, **kamu bisa pakai `props.navigation.navigate`**, lalu **masukan nama screen yang kamu tuju**. Tapi dari mana ya kamu bisa dapat `props.navigation.navigate` ini?



```
// di HTML  
<a href="details.html">Pindah ke detail Screen</a>
```



```
// pada React Native  
props.navigation.navigate('NamaScreenYangAkanDituju')
```



Tenang aja, dapatnya otomatis kok~

Setiap component yang dimasukkan pada **Stack.Screen** secara otomatis akan mendapatkan **props.navigation**, sehingga pada component tersebut kamu bisa pakai **props.navigation** untuk melakukan navigasi antar screen di aplikasimu.

```
function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator initialRouteName="Home">
        <Stack.Screen name="Home" component={HomeScreen} />
        <Stack.Screen name="Details" component={DetailsScreen} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```

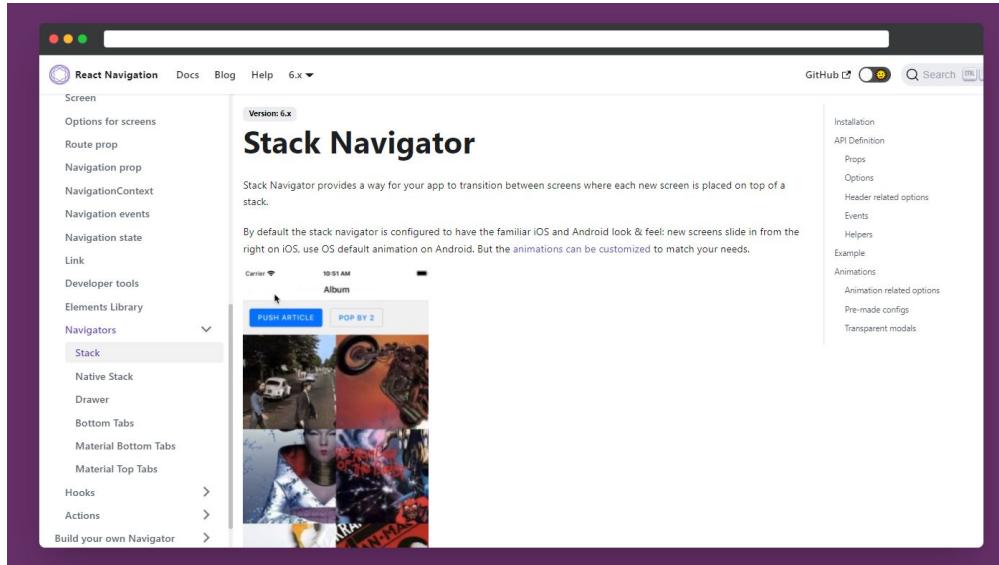
```
function HomeScreen(props) {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Home Screen</Text>
      <Button
        title="Screen Details"
        onPress={() => props.navigation.navigate('Details')}
      />
    </View>
  );
}
```



Horee, selesai Chapter 2 React Native!

Meski begitu, sebenarnya masih banyak hal yang bisa kamu eksplor untuk menambah pengetahuan dan mempertajam skill kamu. Salah satu caranya, baca dokumentasi resmi dari react navigation. Di sana ada banyak step-by-step beserta contoh yang lengkap kalau kamu mau melakukan eksperimen. Langsung aja kepoin link berikut ini, guys!

<https://reactnavigation.org/docs/bottom-tab-navigator>



Saatnya kita Quiz!





1. Pernyataan dibawah ini yang merupakan pernyataan yang benar adalah ...

- A. Kamu bisa menggunakan icon kamu sendiri, tanpa harus menggunakan react-native-icons
- B. Icon di react native terbatas, yang tersedia hanyalah react-native-vector icons
- C. Keduanya benar



1. Pernyataan dibawah ini yang merupakan pernyataan yang benar adalah ...

- A. Kamu bisa menggunakan icon kamu sendiri, tanpa harus menggunakan react-native-icons
- B. Icon di react native terbatas, yang tersedia hanyalah react-native-vector icons
- C. Keduanya benar

Kamu bisa menggunakan custom icon, atau file icon yang telah kamu sediakan tanpa harus menggunakan react-native-vector-icons



2. Apa yang menyebabkan syntax kode di samping apabila dijalankan menjadi error?

- A. Karena, Pemberian props size pada component Icon salah, 30 harusnya nya jadi 30px
- B. Karena, react-native-vector-icons, tidak bisa menerima kode hexadecimal untuk menampilkan warna
- C. Karena, yang di import adalah library react-native-vector-icons nya saja, tapi font nya tidak di import juga

```
● ● ●  
1 import React from 'react';  
2 import { View, Text, StyleSheet } from 'react-native';  
3 import Icon from 'react-native-vector-icons'  
4  
5 const StyleComponent = StyleSheet.create({  
6   textStyle1: {  
7     fontSize: 20,  
8     color: 'red'  
9   },  
10  textStyle2: {  
11    color: 'blue'  
12  }  
13 })  
14  
15 function App(){  
16   <View>  
17     <Text style={[StyleComponent.textStyle1, StyleComponent.textStyle2]}>  
18       Ini Text Mengguanakan style React Native  
19       <Text>  
20         <Icon name="rocket" size={30} color="#900" />  
21     </View>  
22 }
```



2. Apa yang menyebabkan syntax kode di samping apabila dijalankan menjadi error?

- A. Karena, Pemberian props size pada component Icon salah, 30 harusnya nya jadi 30px
- B. Karena, react-native-vector-icons, tidak bisa menerima kode hexadecimal untuk menampilkan warna
- C. Karena, yang di import adalah library react-native-vector-icons nya saja, tapi font nya tidak di import juga

```
● ● ●  
1 import React from 'react';
2 import { View, Text, StyleSheet } from 'react-native';
3 import Icon from 'react-native-vector-icons'
4
5 const StyleComponent = StyleSheet.create({
6   textStyle1: {
7     fontSize: 20,
8     color: 'red'
9   },
10  textStyle2: {
11    color: 'blue'
12 }
13 })
14
15 function App(){
16   <View>
17     <Text style={[StyleComponent.textStyle1, StyleComponent.textStyle2]}>
18       Ini Text Menggunakan style React Native
19       <Text>
20         <Icon name="rocket" size={30} color="#900" />
21       </Text>
22   </View>
23 }
```

Ketika meng import react-native-vector-icons, nama font icon nya harus di import juga



3. Manakah kode yang tepat untuk membuat layout seperti gambar disamping?

A.

```
<View style={{ flex: 1, width: 500, height: 500 }}>
<View style={{ flex: 1, flexGrow: 1 }} />
<View style={{ flex: 1, flexGrow: 2 }} />
<View style={{ flex: 1, flexGrow: 3 }} />
</View>
```

B.

```
<View style={{ flex: 1, width: 500, height: 500 }}>
<View style={{ flex: 1, flexGrow: 1 }} />
<View style={{ flex: 1, flexGrow: 2 }} />
<View style={{ flex: 1, flexGrow: 1 }} />
</View>
```

C.

```
<View style={{ flex: 1, width: 500, height: 500 }}>
<View style={{ flex: 1, flexGrow: 1 }} />
<View style={{ flex: 1, flexGrow: 2 }} />
<View style={{ flex: 1, flexGrow: 4 }} />
</View>
```





3. Manakah kode yang tepat untuk membuat layout seperti gambar disamping?

A.

B.

```
<View style={{ flex: 1, width: 500, height: 500 }}>
  <View style={{ flex: 1, flexGrow: 1 }} />
  <View style={{ flex: 1, flexGrow: 2 }} />
  <View style={{ flex: 1, flexGrow: 1 }} />
</View>
```

C.



Flex-grow adalah properti yang menentukan berapa banyak item akan membesar relatif terhadap sisa item fleksibel di dalam container yang sama.



- 4. Apabila kamu disuruh membuat aplikasi yang memiliki fitur untuk menampilkan list yang berisi banyak data, komponen React Native mana yang tepat digunakan?**
- A. DataSource
 - B. ScrollView
 - C. FlatList



- 4. Apabila kamu disuruh membuat aplikasi yang memiliki fitur untuk menampilkan list yang berisi banyak data, komponen React Native mana yang tepat digunakan?**
- A. DataSource
 - B. ScrollView
 - C. FlatList

FlatList memiliki performa yang lebih baik dibandingkan ScrollView ketika menghandle list data yang banyak.



5. Untuk menampilkan elemen secara horizontal, property yang manakah yang tepat digunakan?

- A. flex: 'row'
- B. flexbox: 'row'
- C. flexDirection: 'row'



5. Untuk menampilkan elemen secara horizontal, property yang manakah yang tepat digunakan?

- A. flex: 'row'
- B. flexbox: 'row'
- C. flexDirection: 'row'

FlexDirection adalah properti yang mengatur aray dari child item pada flexBox, 'row' untuk membuat semua child item menjadi horizontal, 'column' untuk membuat menjadi vertikal

Terima Kasih!



Chapter ✓

completed