

Mobil Programlama

Uygulama 2



```
import { ColorPropType, StyleSheet, Text, View } from 'react-native';
import PropTypes from 'prop-types';
import React from 'react';

export default function Avatar({ size, backgroundColor, initials }) {
  const style = {
    width: size,
    height: size,
    borderRadius: size / 2,
    backgroundColor,
  };

  return (
    <View style={[styles.container, style]}>
      <Text style={styles.text}>{initials}</Text>
    </View>
  );
}

Avatar.propTypes = {
  initials: PropTypes.string.isRequired,
  size: PropTypes.number.isRequired,
  backgroundColor: ColorPropType.isRequired,
};
```

```
Avatar.propTypes = {
  initials: PropTypes.string.isRequired,
  size: PropTypes.number.isRequired,
  backgroundColor: ColorPropType.isRequired,
};

const styles = StyleSheet.create({
  container: {
    alignItems: 'center',
    justifyContent: 'center',
  },
  text: {
    color: 'white',
  },
});
```

```

import Avatar from './Avatar';
import getAvatarColor from '../utils/getAvatarColor';
import getInitials from '../utils/getInitials';

export default function AuthorRow({
  fullname,
  linkText,
  onPressLinkText,
}) {
  return (
    <View style={styles.container}>
      <Avatar
        size={35}
        initials={getInitials(fullname)}
        backgroundColor={getAvatarColor(fullname)}
      />
      <Text style={styles.text} numberOfLines={1}>
        {fullname}
      </Text>
      {!!linkText && (
        <TouchableOpacity onPress={onPressLinkText}>
          <Text numberOfLines={1}>{linkText}</Text>
        </TouchableOpacity>
      )}
    </View>
  );
}

```

```

AuthorRow.propTypes = {
  fullname: PropTypes.string.isRequired,
  linkText: PropTypes.string.isRequired,
  onPressLinkText: PropTypes.func.isRequired,
};

const styles = StyleSheet.create({
  container: {
    height: 50,
    flexDirection: 'row',
    alignItems: 'center',
    paddingHorizontal: 10,
  },
  text: {
    flex: 1,
    marginHorizontal: 6,
  },
});

```

```
export default function getAvatarColor(name) {  
  const hexCode = name  
    .split('')  
    .reduce((acc, char) => (acc * char.charCodeAt(0)) % 0xffffffff, 1)  
    .toString(16);  
  
  return `#${'0'.repeat(6 - hexCode.length) + hexCode}`;  
}
```

```
export default function getInitials(fullname) {  
  const match = fullname.match(/(\w)?\w*\s*(\w)?/);  
  return match ? match.slice(1).join('') : '';  
}
```

```

import {ActivityIndicator, Image, StyleSheet, View, } from 'react-native';
import PropTypes from 'prop-types';
import React from 'react';

import AuthorRow from './AuthorRow';

export default class Card extends React.Component {
  static propTypes = {
    fullname: PropTypes.string.isRequired,
    image: Image.propTypes.source.isRequired,
    linkText: PropTypes.string.isRequired,
    onPressLinkText: PropTypes.func.isRequired,
  };

  state = {
    loading: true,
  };

  handleLoad = () => {
    this.setState({ loading: false });
  };

  render() {
    const { fullname, image, linkText, onPressLinkText } = this.props;
    const { loading } = this.state;

```

```

    return (
      <View>
        <AuthorRow
          fullname={fullname}
          linkText={linkText}
          onPressLinkText={onPressLinkText}
        />
        <View style={styles.image}>
          {loading && (
            <ActivityIndicator
              style={StyleSheet.absoluteFill}
              size={'large'}
            />
          )}
          <Image
            style={StyleSheet.absoluteFill}
            source={image}
            onLoad={this.handleLoad}
          />
        </View>
      </View>
    );
  }
}

const styles = StyleSheet.create({
  image: {
    aspectRatio: 1,
    backgroundColor: 'rgba(0,0,0,0.02)',
  },
});

```

```

import { FlatList } from 'react-native';
import PropTypes from 'prop-types';
import React from 'react';

import { getImageFromId } from '../utils/api';
import Card from './Card';

const keyExtractor = ({ id }) => id.toString();

export default class CardList extends React.Component {
  static propTypes = {
    items: PropTypes.arrayOf(
      PropTypes.shape({
        id: PropTypes.number.isRequired,
        author: PropTypes.string.isRequired,
      }).isRequired,
    ).isRequired,
  };

```

```

export const fetchImages = async () => {
  const response = await fetch('https://unsplash.it/list');
  const images = await response.json();

  return images;
};

```

```

export const getImageFromId = id =>
  `https://unsplash.it/${600}/${600}?image=${id}`;

```

```

renderItem = ({ item: { id, author } }) => (
  <Card
    fullname={author}
    image={{
      uri: getImageFromId(id),
    }}
  />
);

render() {
  const { items } = this.props;

  return (
    <FlatList
      data={items}
      renderItem={this.renderItem}
      keyExtractor={keyExtractor}
    />
  );
}

```

Api.js

```
export const fetchImages = async () => {  
  const response = await fetch('https://unsplash.it/list');  
  const images = await response.json();  
  
  return images;  
};  
  
export const getImageFromId = id =>  
  `https://unsplash.it/${600}/${600}?image=${id}`;
```