

React Native



Conceptos Básicos

Propiedades

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

class MyText extends React.Component {
  render() {
    return (
      <Text style={{color: this.props.color,fontSize: 30}}>
        {this.props.value}!
      </Text>
    )
  }
}

export default class App extends React.Component {
  render() {
    return (
      <View style={{alignItems: 'center'}}>
        <MyText color='blue' value='Hello World' />
        <MyText color='black' value='Goodbye World' />
      </View>
    )
  }
}
```



Estado

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

class MyText extends React.Component {
  constructor(props) {
    super(props);
    this.state = {color: this.props.initialColor};
    setInterval(() => {
      this.setState({color: (this.state.color == 'blue' ? 'black' : 'blue')});
    }, 1000);
  }
  render() {
    return (
      <Text style={{color: this.state.color, fontSize: 30}}>{this.props.value}</Text>
    )
  }
}

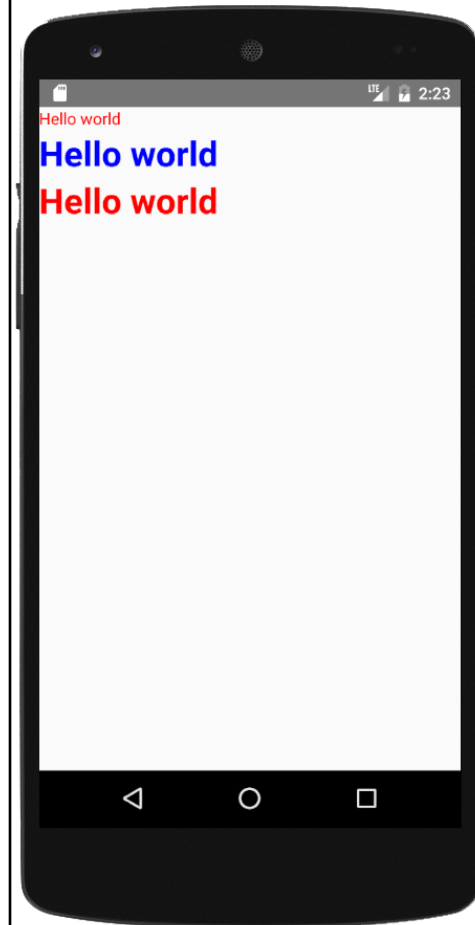
export default class App extends React.Component {
  render() {
    return (
      <View style={{alignItems: 'center'}}>
        <MyText initialColor='blue' value='Hello World' />
        <MyText initialColor='black' value='Goodbye World' />
      </View>
    )
  }
}
```



Estilos

```
import React from 'react';
import { StyleSheet, Text, View } from 'react-native';
export default class App extends React.Component {
  render() {
    return (
      <View>
        <Text style={styles.red}>Hello world</Text>
        <Text style={styles.bigblue}>Hello world</Text>
        <Text style={[styles.bigblue, styles.red]}>Hello world</Text>
      </View>
    )
  }
}

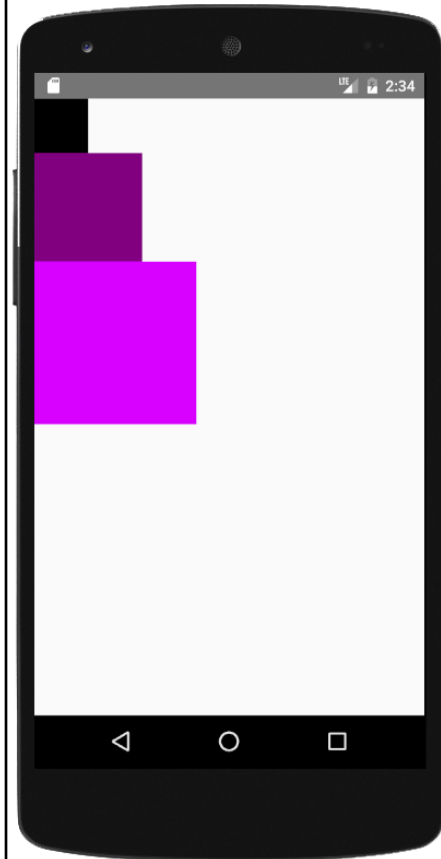
const styles = StyleSheet.create({
  red: {
    color: 'red',
  },
  bigblue: {
    color: 'blue',
    fontWeight: 'bold',
    fontSize: 30
  }
})
```



Dimensiones fijas: Width and Height

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

export default class App extends React.Component {
  render() {
    return (
      <View>
        <View style={{width: 50, height: 50, backgroundColor: 'rgb(0, 0, 0)}} />
        <View style={{width: 100, height: 100, backgroundColor: 'purple'}} />
        <View style={{width: 150, height: 150, backgroundColor: '#d800ff'}} />
      </View>
    )
  }
}
```



En React Native, todas las dimensiones representan *density independent pixels* (**dp/dip**)

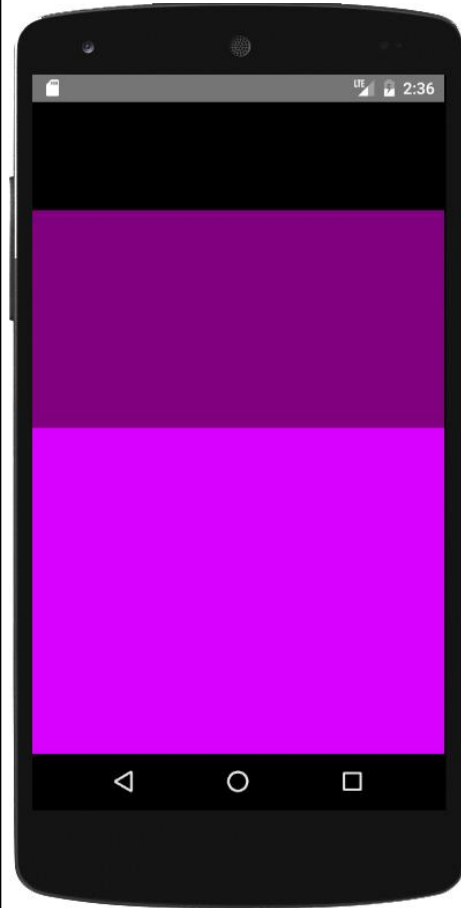
Pantallas: Densidad y píxeles

- **Pixel:** la superficie homogénea en color más pequeña que forma parte de una imagen digital.
- **Densidad de pantalla:** la cantidad de píxeles en un área física de la pantalla. Normalmente se representa como **dpi** (dots per inch) / ppp (puntos por pulgada).
- **Resolución de pantalla.** Cantidad total de píxeles físicos.
- **Píxeles independientes de la densidad (dp/dip).**
Es una unidad de pixel virtual para expresar **dimensiones o posiciones** en una pantalla de una manera independiente de la densidad.
1 dp equivale a 1 pixel físico de una pantalla de 160dpi
$$px = dp * (dpi / 160), dp = px * (160 / dpi)$$
- Más información:
https://developer.android.com/guide/practices/screens_support.html

Dimensiones fluidas: Flexbox

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

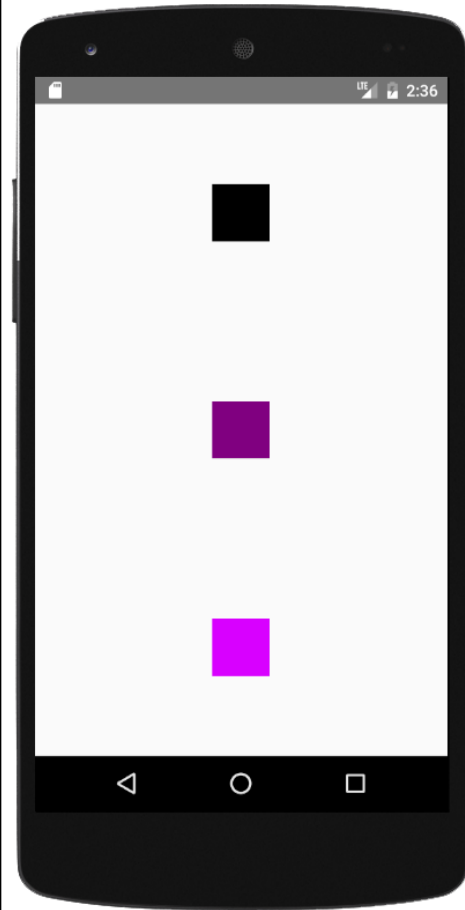
export default class App extends React.Component {
  render() {
    return (
      <View style={{flex: 1}}>
        <View style={{flex: 1, backgroundColor: 'rgb(0, 0, 0)}} />
        <View style={{flex: 2, backgroundColor: 'purple'}} />
        <View style={{flex: 3, backgroundColor: '#d800ff'}} />
      </View>
    )
  }
}
```



Layout con Flexbox

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

export default class App extends React.Component {
  render() {
    return (
      <View style={{
        flex: 1,
        flexDirection: 'column',
        justifyContent: 'space-around',
        alignItems: 'center',
      }}>
        <View style={{width: 50, height: 50, backgroundColor: 'rgb(0, 0, 0)}} />
        <View style={{width: 50, height: 50, backgroundColor: 'purple'}} />
        <View style={{width: 50, height: 50, backgroundColor: '#d800ff'}} />
      </View>
    )
  }
}
```

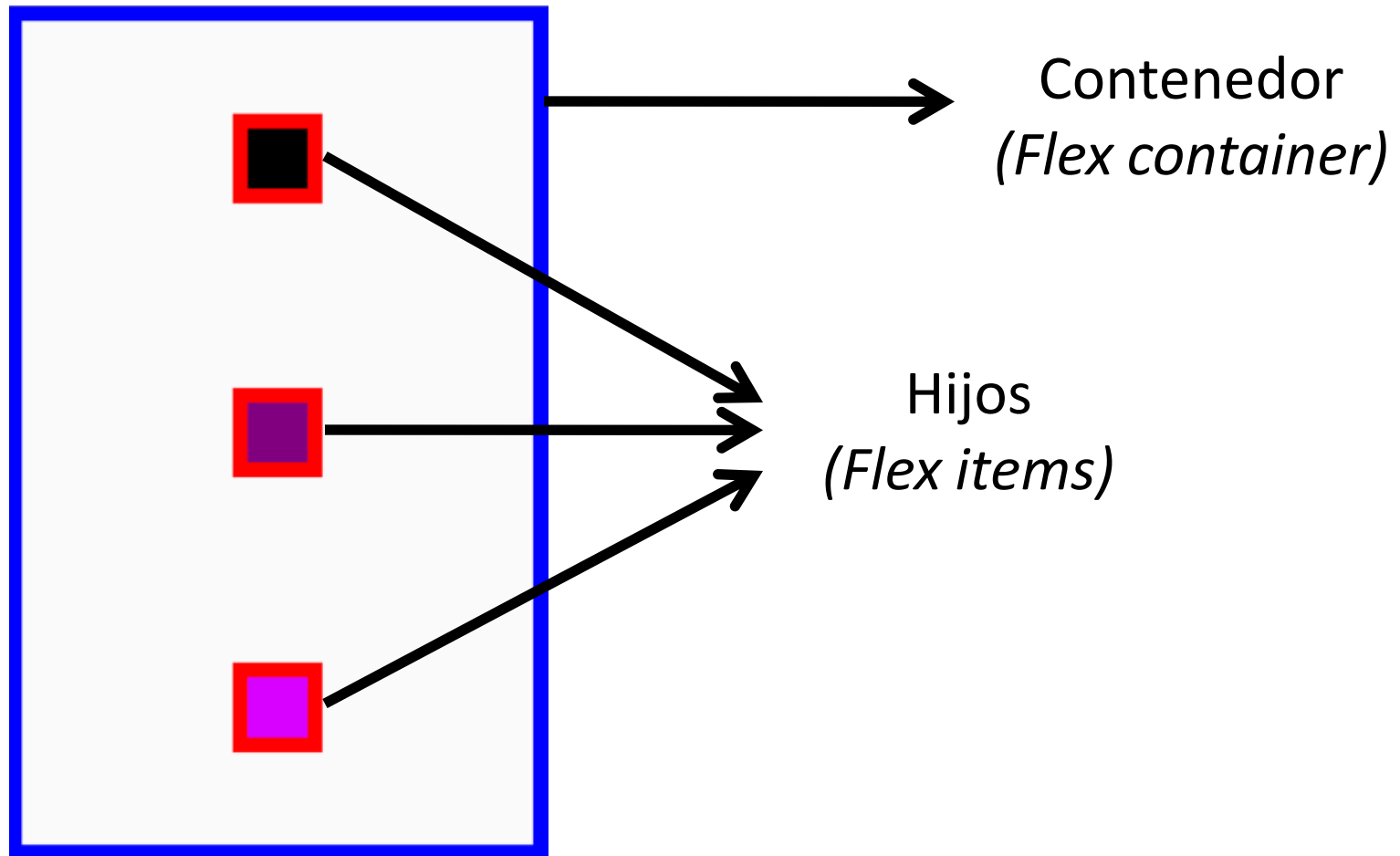


Flexbox

- **Flexbox** es un nuevo modo de layout introducido en **CSS3**
- En **React Native Flexbox** funciona **prácticamente igual que en CSS** aunque con algunas excepciones
- Asegura que los elementos se comportan de manera predecible cuando el layout de la aplicación debe adaptarse a **diferentes tamaños de pantalla**
- No usa elementos flotantes (*float*)
- Más información:
 - http://www.w3schools.com/css/css3_flexbox.asp
 - <https://css-tricks.com/snippets/css/a-guide-to-flexbox>
 - <https://facebook.github.io/react-native/docs/flexbox.html>

Flexbox

- Un **contenedor** (o componente) puede especificar el **layout** de sus **hijos** utilizando **Flexbox**



Flexbox: Atributos de los contenedores

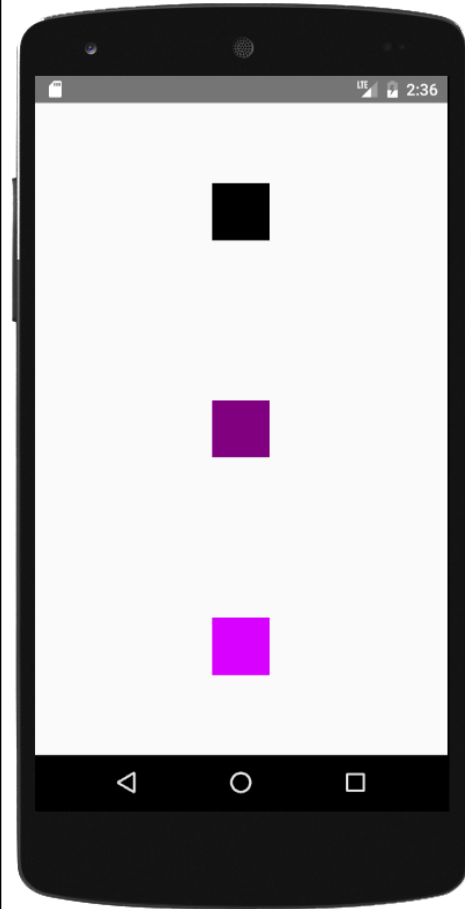
- Un **contenedor** (o componente) puede especificar el **layout** de sus **hijos** utilizando **Flexbox**
- Generalmente se usa una combinación de **3** atributos:
 - **flexDirection**: determina el eje primario del layout
Los hijos se pueden organizar:
 - **row**: horizontalmente
 - **column**: verticalmente
 - **justifyContent**: determina el **posicionamiento** de los hijos a lo largo del **eje primario**
 - **alignItems**: determina el **posicionamiento** de los hijos a lo largo del **eje secundario** (row si el primario es column y viceversa)

Flexbox: flexDirection

Determina el **eje primario** del layout

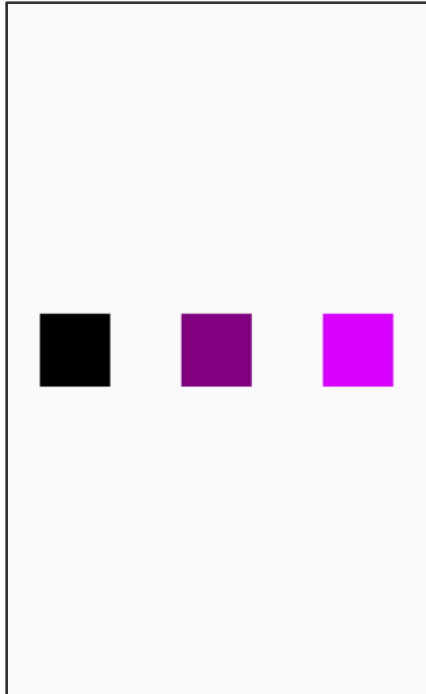
```
import React from 'react';
import { View } from 'react-native';

export default class App extends React.Component {
  render() {
    return (
      <View style={{
        flex: 1,
        flexDirection: 'column',
        justifyContent: 'space-around',
        alignItems: 'center',
      }}>
        <View style={{width: 50, height: 50, backgroundColor: 'rgb(0, 0, 0)}} />
        <View style={{width: 50, height: 50, backgroundColor: 'purple'}} />
        <View style={{width: 50, height: 50, backgroundColor: '#d800ff'}} />
      </View>
    )
  }
}
```

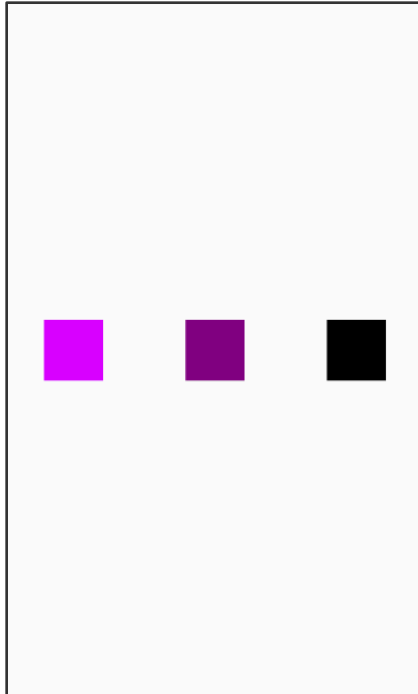


Flexbox: flexDirection

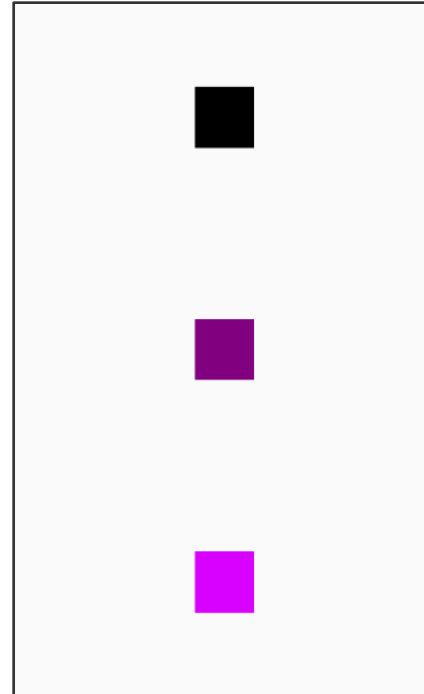
Determina el **eje primario** del layout



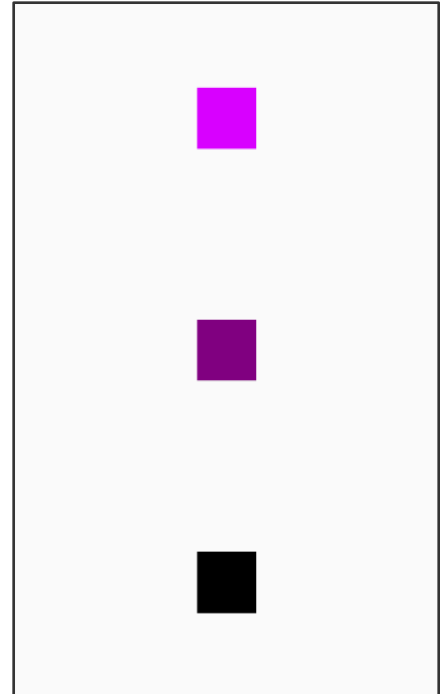
flexDirection:
'row'



flexDirection:
'row-reverse'



flexDirection:
'column'



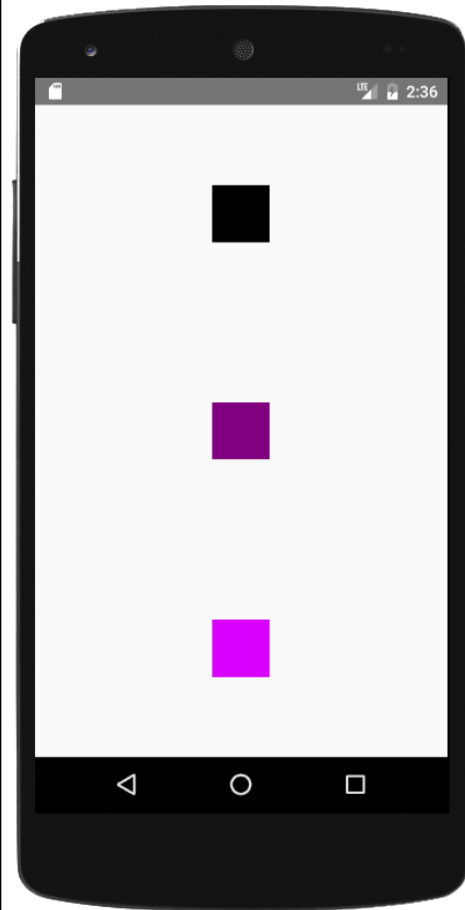
flexDirection:
'column-reverse'

Flexbox: justifyContent

Determina el **posicionamiento** de los hijos a lo largo del **eje primario**






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

export default class App extends React.Component {
  render() {
    return (
      <View style={{
        flex: 1,
        flexDirection: 'column',
        justifyContent: 'space-around',
        alignItems: 'center',
      }}>
        <View style={{width: 50, height: 50, backgroundColor: 'rgb(0, 0, 0)}} />
        <View style={{width: 50, height: 50, backgroundColor: 'purple'}} />
        <View style={{width: 50, height: 50, backgroundColor: '#d800ff'}} />
      </View>
    )
  }
}
```



Flexbox: justifyContent

Determina el **posicionamiento** de los hijos a lo largo del **eje primario**

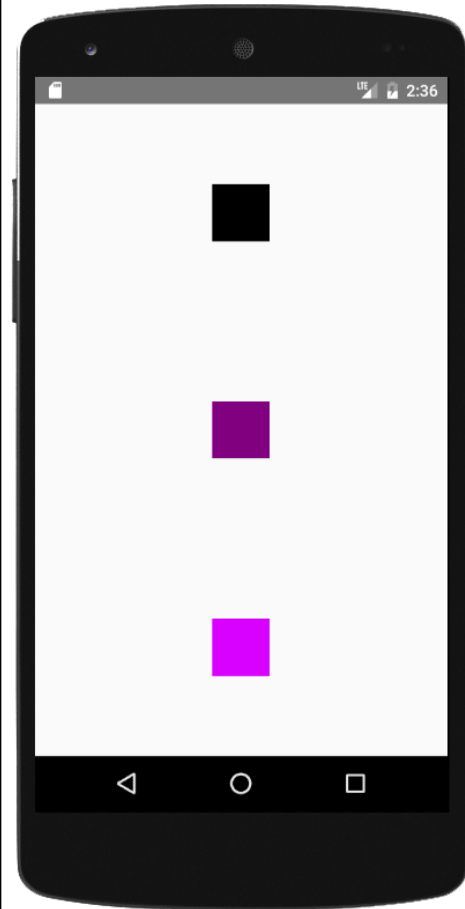
| | | | | |
|---|---|--|---|--|
|  |  |  |  |  |
| justifyContent: 'flex-start' | justifyContent: 'center' | justifyContent: 'flex-end' | justifyContent: 'space-around' | justifyContent: 'space-between' |

Flexbox: alignItems

Determina el **posicionamiento** de los hijos a lo largo del **eje secundario**

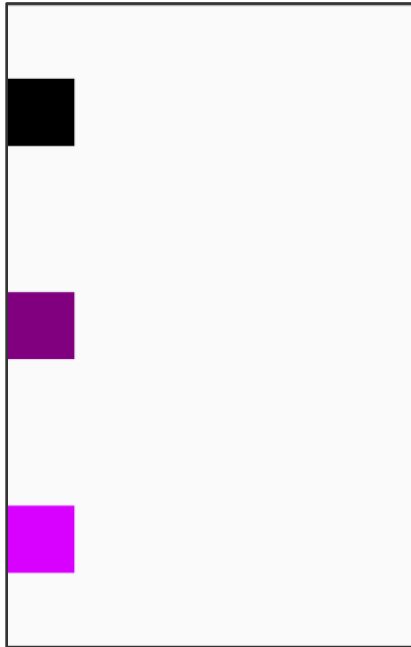
```
import React from 'react';
import { View } from 'react-native';

export default class App extends React.Component {
  render() {
    return (
      <View style={{
        flex: 1,
        flexDirection: 'column',
        justifyContent: 'space-around',
        alignItems: 'center',
      }}>
        <View style={{width: 50, height: 50, backgroundColor: 'rgb(0, 0, 0)}} />
        <View style={{width: 50, height: 50, backgroundColor: 'purple'}} />
        <View style={{width: 50, height: 50, backgroundColor: '#d800ff'}} />
      </View>
    )
  }
}
```

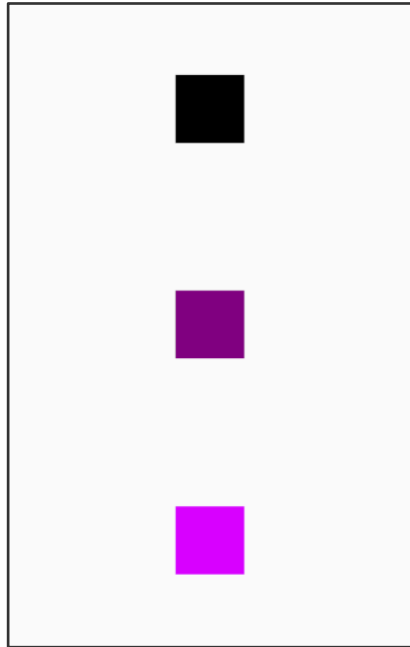


Flexbox: alignItems

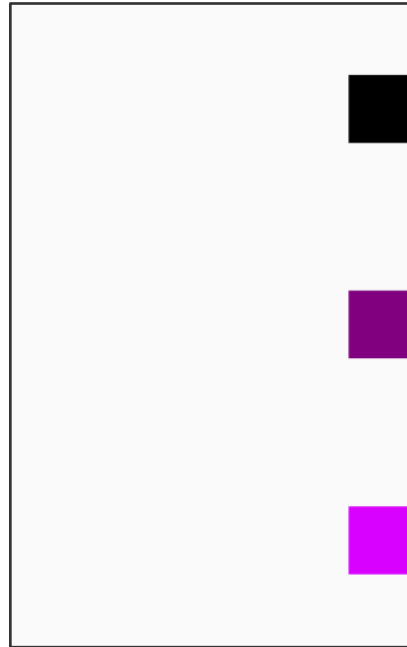
Determina el **posicionamiento** de los hijos a lo largo del **eje secundario**



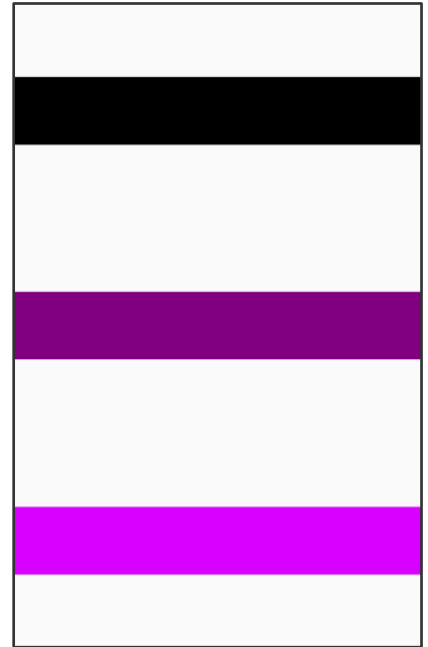
alignItems:
'flex-start'



alignItems:
'center'



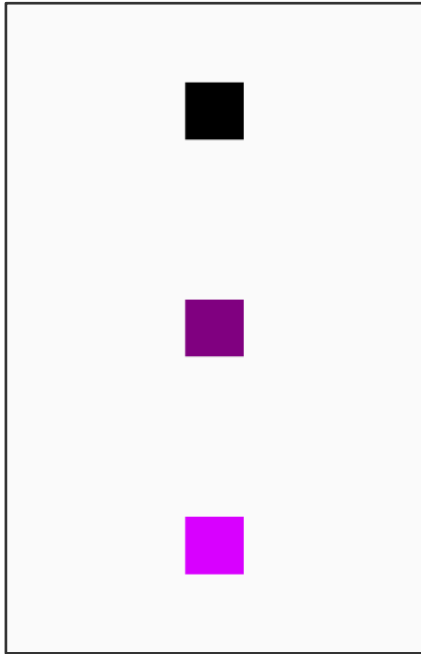
alignItems:
'flex-end'



alignItems:
'stretch'

Para que **stretch** tenga efecto, los hijos no deben tener una dimensión fija en el eje secundario

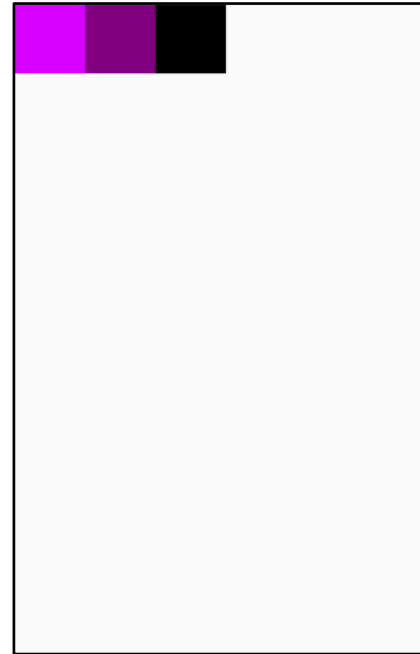
Flexbox: Ejemplos



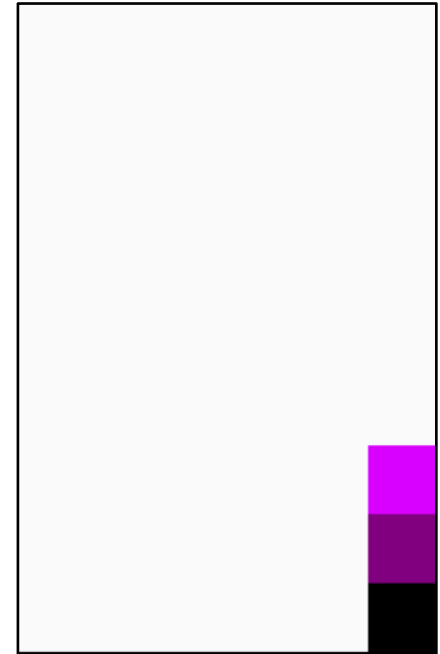
```
{  
  flex: 1,  
  flexDirection:  
    'column',  
  justifyContent:  
    'space-around',  
  alignItems:  
    'center'  
}
```



```
{  
  flex: 1,  
  flexDirection:  
    'row',  
  justifyContent:  
    'flex-end',  
  alignItems:  
    'flex-start'  
}
```



```
{  
  flex: 1,  
  flexDirection:  
    'row-reverse',  
  justifyContent:  
    'flex-end',  
  alignItems:  
    'flex-start'  
}
```



```
{  
  flex: 1,  
  flexDirection:  
    'column-reverse',  
  justifyContent:  
    'flex-start',  
  alignItems:  
    'flex-end'  
}
```

Flexbox: Atributos de los items

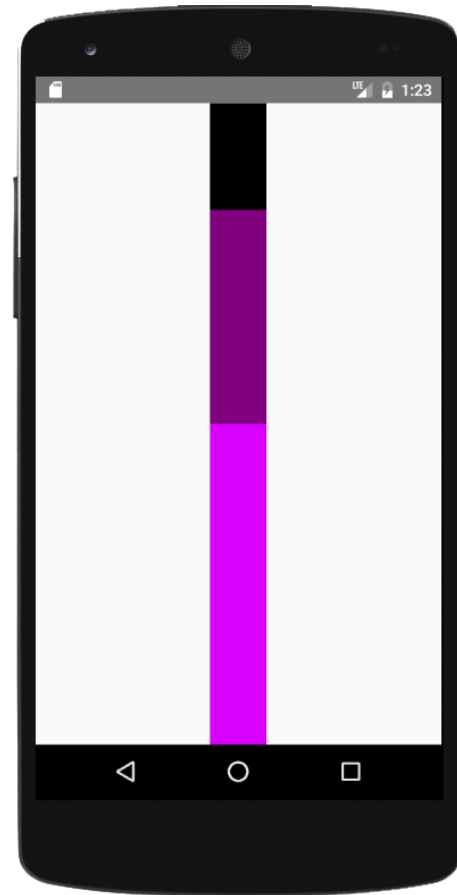
- Los hijos de los contenedores (*flexbox items*) pueden especificar ciertos atributos:
 - **flex:** especifica el tamaño del item en el eje primario en relación al resto de items del mismo contenedor
 - **alignSelf:** sobrescribe el atributo 'alignItems' del contenedor para este item
 - **margin y padding:** los atributos de Flexbox se pueden combinar con otros atributos como margin y padding

Flexbox: flex

Especifica el tamaño del item en el eje primario en relación al resto de items del mismo contenedor

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

export default class App extends React.Component {
  render() {
    return (
      <View style={{
        flex: 1,
        flexDirection: 'column',
        justifyContent: 'space-around',
        alignItems: 'center',
      }}>
        <View style={{flex: 1, width: 50, backgroundColor: 'rgb(0, 0, 0)}} />
        <View style={{flex: 2, width: 50, backgroundColor: 'purple'}} />
        <View style={{flex: 3, width: 50, backgroundColor: '#d800ff'}} />
      </View>
    )
  }
}
```

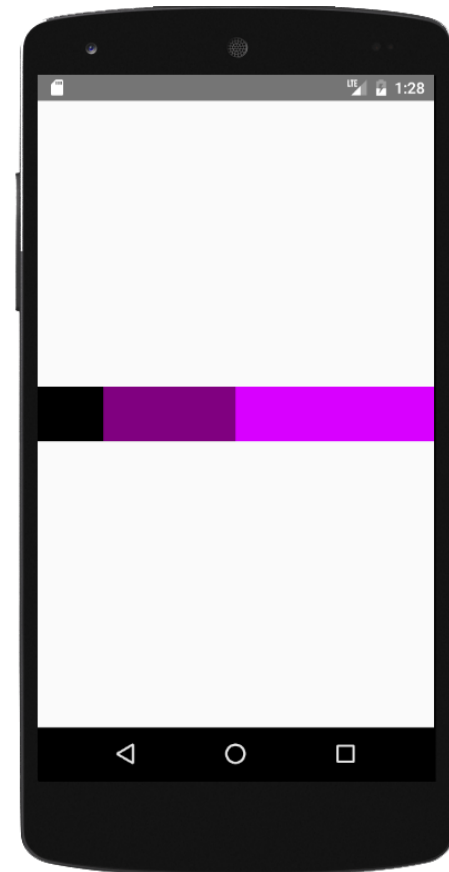


Flexbox: flex

Especifica el tamaño del item en el eje primario en relación al resto de items del mismo contenedor

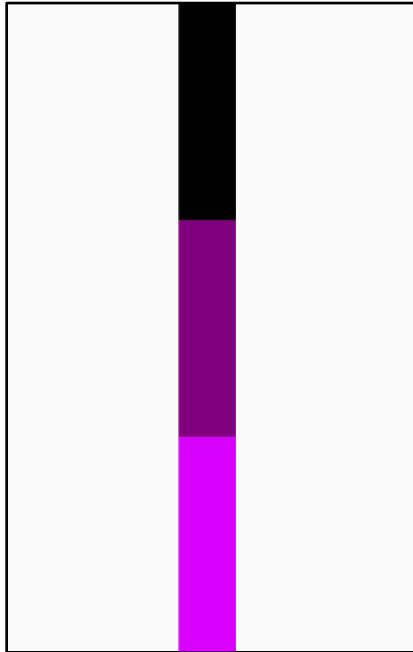
```
import React from 'react';
import { View } from 'react-native';

export default class App extends React.Component {
  render() {
    return (
      <View style={{
        flex: 1,
        flexDirection: 'row',
        justifyContent: 'space-around',
        alignItems: 'center',
      }}>
        <View style={{flex: 1, height: 50, backgroundColor: 'rgb(0, 0, 0)}} />
        <View style={{flex: 2, height: 50, backgroundColor: 'purple'}} />
        <View style={{flex: 3, height: 50, backgroundColor: '#d800ff'}} />
      </View>
    )
  }
}
```

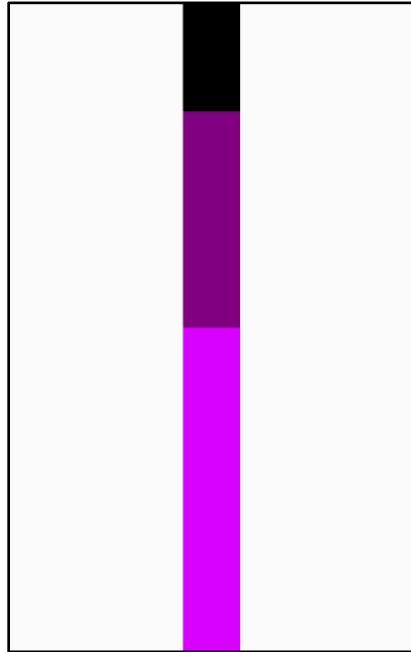


Flexbox: flex

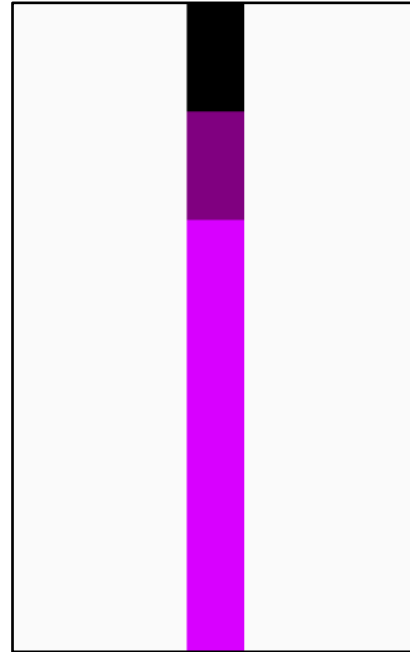
Especifica el tamaño del item en el eje primario en relación al resto de items del mismo contenedor



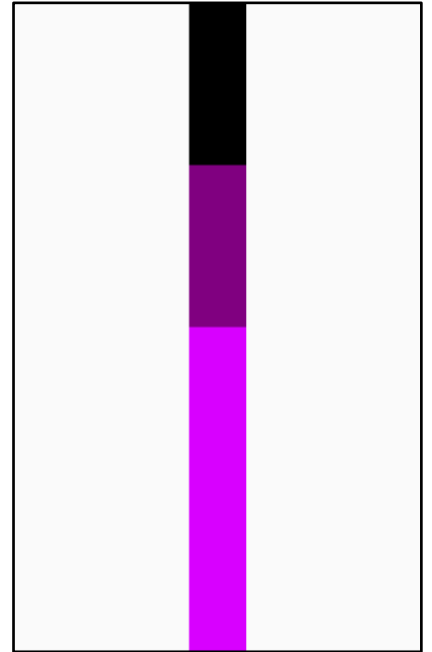
```
<View style={{  
  flex: 1, ...}} />  
<View style={{  
  flex: 1, ...}} />  
<View style={{  
  flex: 1, ...}} />
```



```
<View style={{  
  flex: 1, ...}} />  
<View style={{  
  flex: 2, ...}} />  
<View style={{  
  flex: 3, ...}} />
```



```
<View style={{  
  flex: 1, ...}} />  
<View style={{  
  flex: 1, ...}} />  
<View style={{  
  flex: 4, ...}} />
```



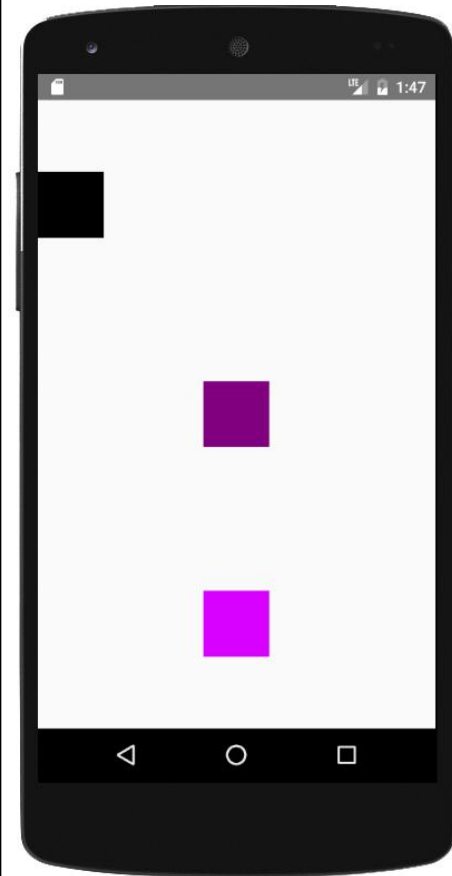
```
<View style={{  
  flex: 2, ...}} />  
<View style={{  
  flex: 2, ...}} />  
<View style={{  
  flex: 4, ...}} />
```

Flexbox: alignSelf

Sobrescribe el atributo 'alignItems' del contenedor para este item

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

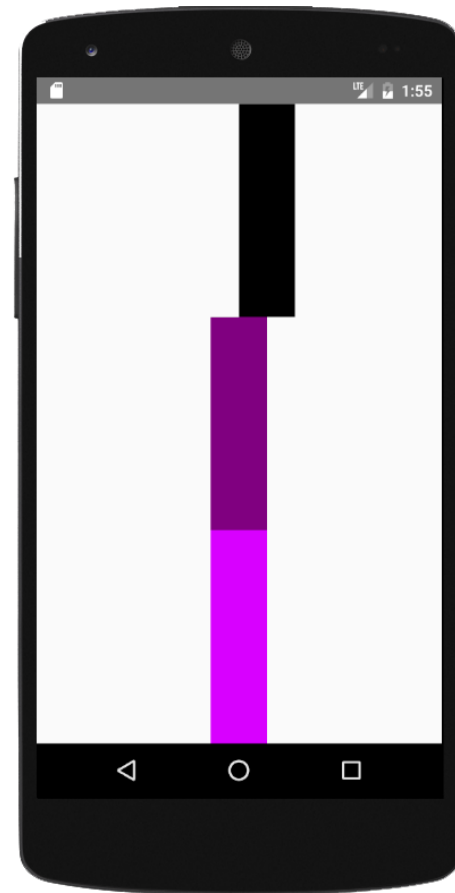
export default class App extends React.Component {
  render() {
    return (
      <View style={{
        flex: 1,
        flexDirection: 'column',
        justifyContent: 'space-around',
        alignItems: 'center',
      }}>
        <View style={{alignSelf: 'flex-start', width: 50, height: 50,
          backgroundColor: 'rgb(0, 0, 0)'} />
        <View style={{width: 50, height: 50, backgroundColor: 'purple'}} />
        <View style={{width: 50, height: 50, backgroundColor: '#d800ff'}} />
      </View>
    )
  }
}
```



Flexbox: Uso con margin

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

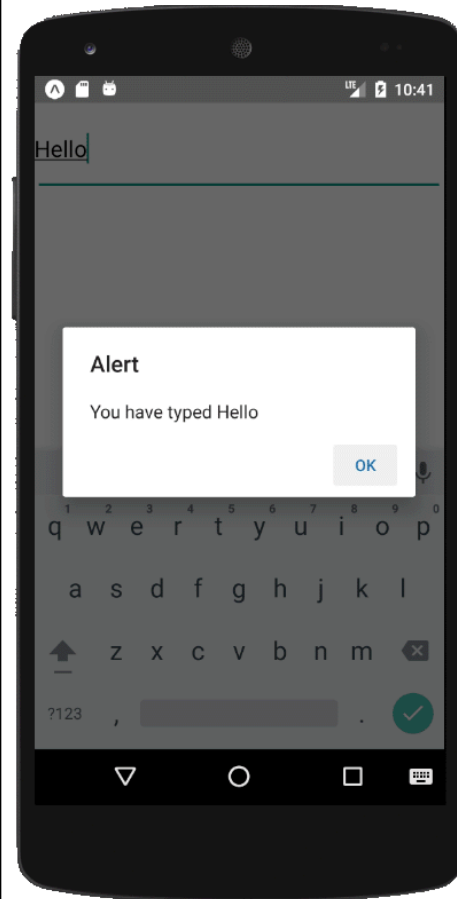
export default class App extends React.Component {
  render() {
    return (
      <View style={{
        flex: 1,
        flexDirection: 'column',
        justifyContent: 'space-around',
        alignItems: 'center',
      }}>
        <View style={{marginLeft: 50, flex: 1, width: 50,
          backgroundColor: 'rgb(0, 0, 0)}} />
        <View style={{flex:1, width: 50, backgroundColor: 'purple'}} />
        <View style={{flex:1, width: 50, backgroundColor: '#d800ff'}} />
      </View>
    )
  }
}
```



TextInput

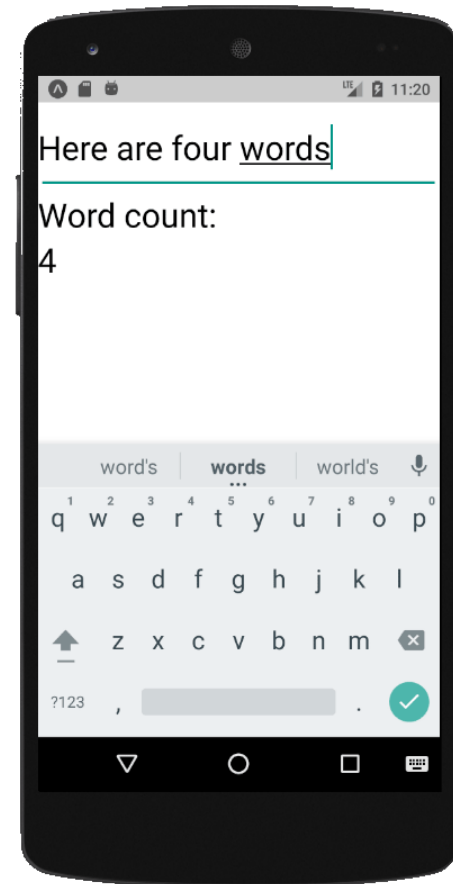
```
import React from 'react';
import { TextInput, View } from 'react-native';

export default class App extends React.Component {
  _onTextInputChange(text){
    if(text === "Hello"){
      alert("You have typed Hello")
    }
  }
  render() {
    return (
      <View>
        <TextInput style={{height: 80, fontSize: 20}}
          placeholder= "Type Hello"
          onChangeText={this._onTextInputChange} />
      </View>
    )
  }
}
```



Text y TextInput

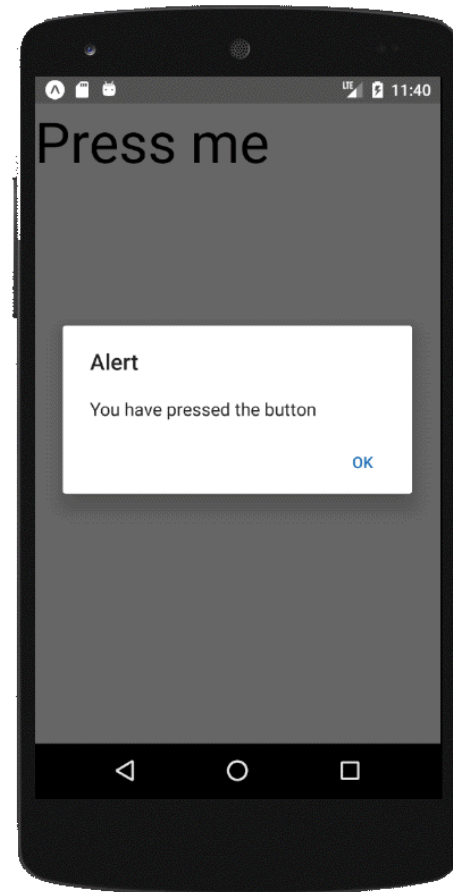
```
import React from 'react';
import { Text, TextInput, View } from 'react-native';
export default class App extends React.Component {
  constructor(props){
    super(props);
    this.state = {value: ""};
  }
  _onTextInputChange(text){
    this.setState({value: text});
  }
  render() {
    return (
      <View>
        <TextInput style={{height: 80, fontSize: 30}} placeholder="Type here"
          onChangeText={this._onTextInputChange.bind(this)} />
        <Text style={{fontSize: 30}}>Word count:</Text>
        <Text style={{fontSize: 30}}>
          {this.state.value.split(/\s+/).filter((w) => w!="").length}
        </Text>
      </View>
    )
  }
}
```



TouchableHighlight

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

export default class App extends React.Component {
  _onPressButton(){
    alert("You have pressed the button")
  }
  render() {
    return (
      <View>
        <TouchableHighlight onPress={this._onPressButton}>
          <Text style={{fontSize: 50}}>Press me</Text>
        </TouchableHighlight>
      </View>
    )
  }
}
```

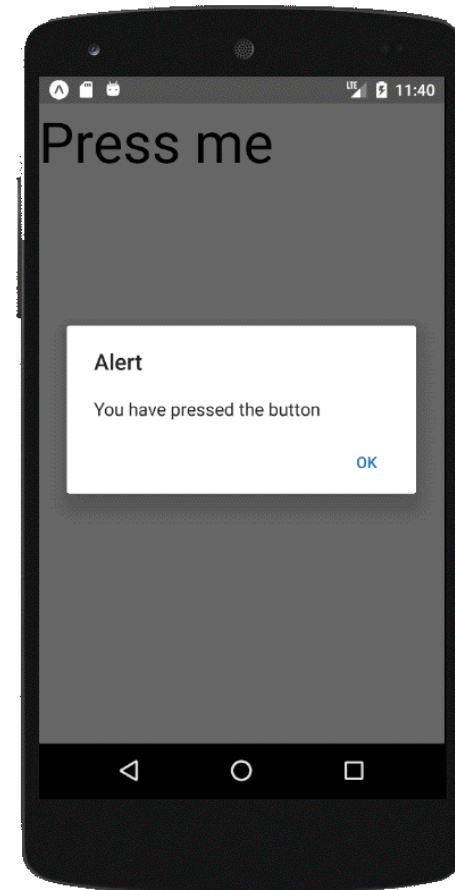


Componente MyButton

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

class MyButton extends React.Component {
  render(){
    return (
      <TouchableHighlight onPress={this.props.onClick}>
        <Text style={{fontSize: 50}}>{this.props.text}</Text>
      </TouchableHighlight>
    )
  }
}

export default class App extends React.Component {
  render() {
    return (
      <MyButton
        text="Press me"
        onClick={() => alert("You have pressed the button")}>
      </MyButton>
    )
  }
}
```



Image

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

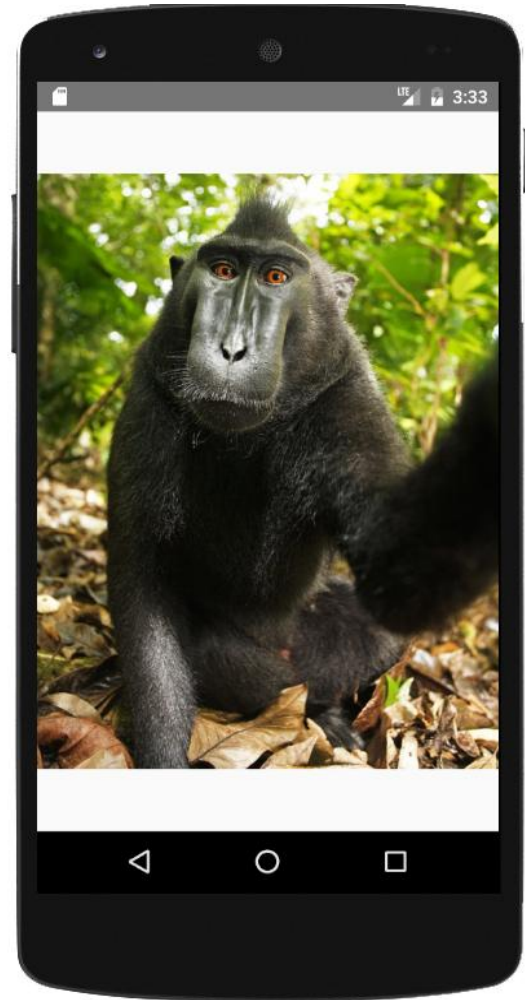
export default class App extends React.Component {
  render() {
    return (
      <View style={{
        flex:1,
        alignItems: 'center',
        justifyContent: 'center'
      }}>
        <Image source={require('./img/macaca_nigra.jpg')} />
      </View>
    )
  }
}
```



Image

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

export default class App extends React.Component {
  render() {
    return (
      <View style={{
        flex:1
      }}>
        <Image
          resizeMode='contain'
          style={{flex:1}}
          source={{uri: "https://vishub.org/pictures/8490.jpg"}}
        />
      </View>
    )
  }
}
```



ScrollView

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

export default class App extends React.Component {
  render() {
    var images = [];
    for(let i=0; i<30; i++){
      images.push(
        <Image resizeMode='contain'
          style={{width:100, height: 100}}
          source={require('./img/macaca_nigra.jpg')} key={i} />
      )
    }
    return (
      <View style={{height: 500}}>
        <ScrollView>{images}</ScrollView>
      </View>
    )
  }
}
```



ListViews

- React Native proporciona varios componentes para presentar listas de datos
 - **FlatList** (<https://facebook.github.io/react-native/docs/flatlist.html>)
 - Permite mostrar de forma eficiente datos similarmente estructurados en listas con scroll
 - **SectionList** (<https://facebook.github.io/react-native/docs/sectionlist.html>)
 - Permite mostrar conjuntos de datos en listas con diferentes secciones
 - Para crear listas sin secciones mejor usar FlatList
 - **~~ListView~~ (Obsoleto)**

Tras crear una lista con FlatList, SectionList o ListView se pueden modificar los datos de la misma

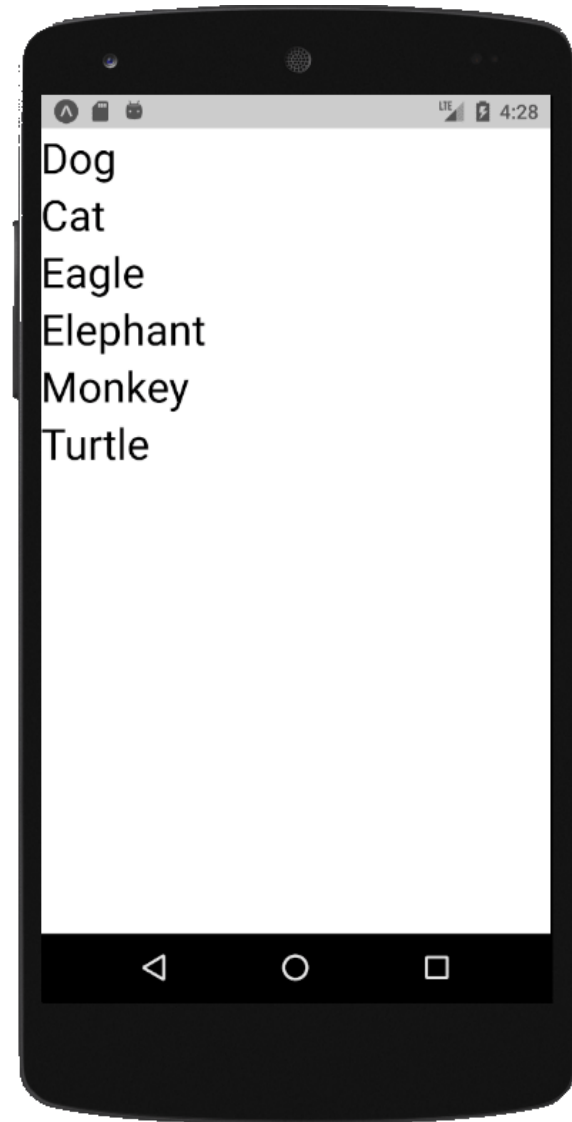
FlatList

- **FlatList** permite mostrar de forma eficiente datos similarmente estructurados en listas verticales u horizontales con scroll
- Los datos se pueden modificar una vez creada la lista
- Componente pensado para grandes listas de datos:
FlatList solo renderiza aquellos elementos que se muestran por pantalla mientras que *ScrollView* renderiza todos los elementos
- FlatList requiere dos propiedades:
 - **data**: fuente de información de la lista
 - **renderItem**: función que recibe como parámetro un elemento de la lista y devuelve un componente renderizable
- Documentación:
<https://facebook.github.io/react-native/docs/using-a-listview.html>
<https://facebook.github.io/react-native/docs/flatlist.html>

FlatList

```
import React from 'react';
import { Text, FlatList } from 'react-native';

export default class App extends React.Component {
  render() {
    var animals = [{key:'Dog'}, {key:'Cat'}, {key:'Eagle'},
      {key:'Elephant'}, {key:'Monkey'}, {key:'Turtle'}];
    return (
      <FlatList
        data={animals}
        renderItem={({item}) =>
          <Text style={{fontSize: 30}}>{item.key}</Text>
        />
      )
    )
  }
}
```

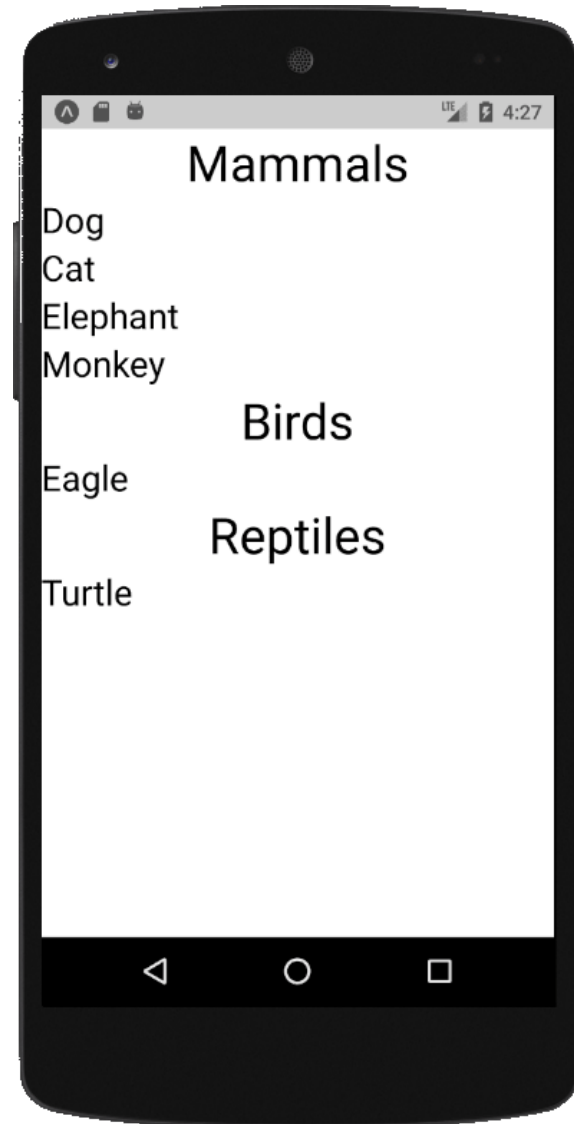


SectionList

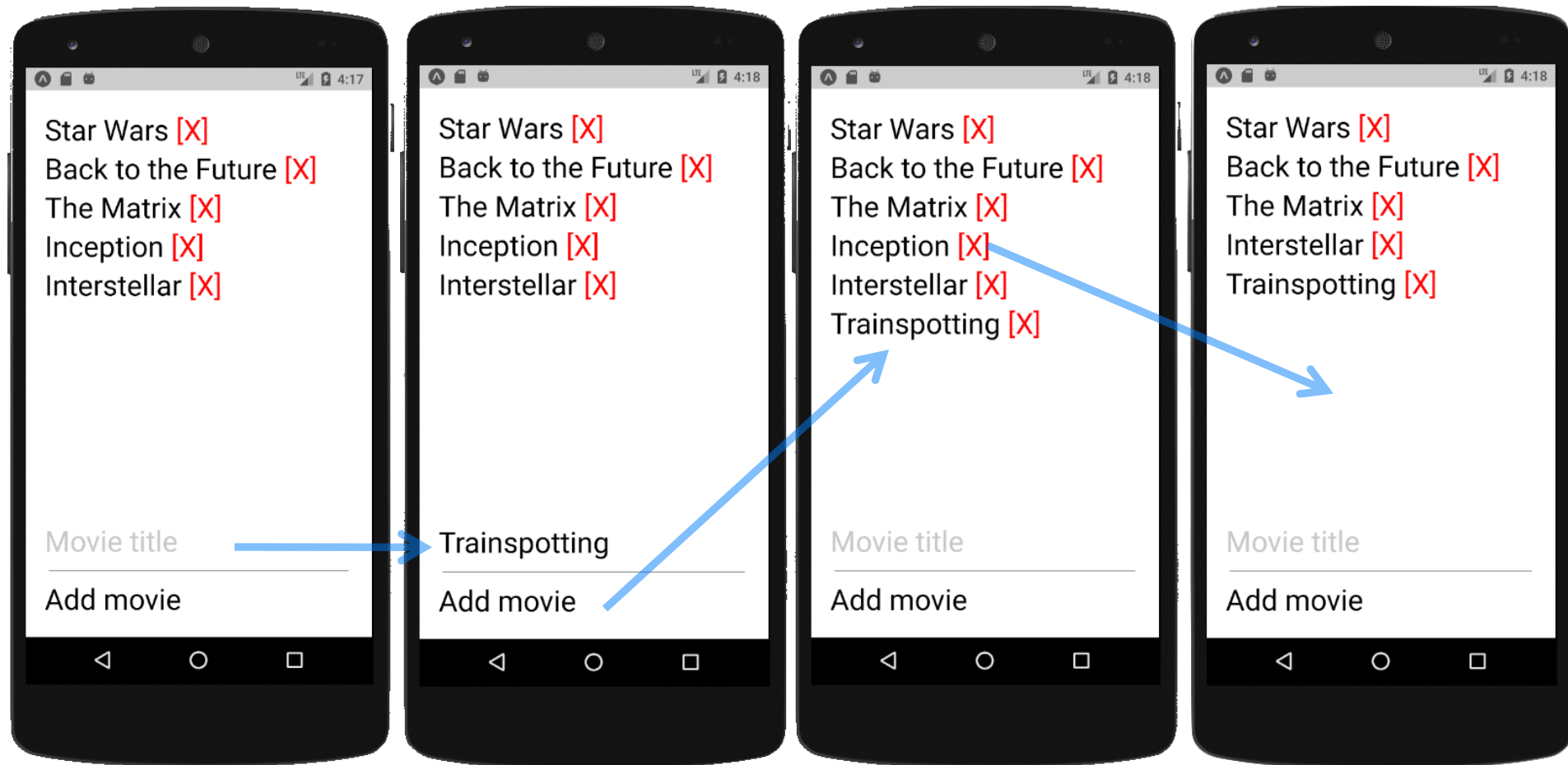
```
import React from 'react';
import { Text, SectionList } from 'react-native';

export default class App extends React.Component {
  render() {
    var mammals = [{key:'Dog'}, {key:'Cat'}, {key:'Elephant'},
                    {key:'Monkey'}];

    var birds = [{key:'Eagle'}];
    var reptiles = [{key:'Turtle'}];
    return (
      <SectionList
        sections={[
          {data: mammals, title: 'Mammals'},
          {data: birds, title: 'Birds'},
          {data: reptiles, title: 'Reptiles'}
        ]}
        renderItem={({item}) =>
          <Text style={{fontSize:25}}>{item.key}</Text>
        }
        renderSectionHeader={({section}) =>
          <Text style={{textAlign:'center', fontSize:35}}>
            {section.title}</Text>
          </Text>
        )
      />
    )
  }
}
```



Ejemplo: Lista dinámica con FlatList



Ejemplo: Lista dinámica con FlatList

```
import React from 'react';
import { Text, TextInput, TouchableHighlight, View, FlatList } from 'react-native';

export default class App extends React.Component {
  [...]
  render() {
    return (
      <View style={{flex: 1, padding: 20}}>
        <FlatList
          data={this.state.movies}
          renderItem={this._renderItem.bind(this)}
        />
        <TextInput
          style={{height: 80, fontSize: 30}} placeholder='Movie title'
          value={this.state.valueTextInput}
          onChangeText={({text}) => this.setState({valueTextInput: text})} />
        <TouchableHighlight onPress={this._addMovieFromTextInput.bind(this)}>
          <Text style={{fontSize: 30}}>Add movie</Text>
        </TouchableHighlight>
      </View>
    )
  }
}
```

Ejemplo: Lista dinámica con FlatList

```
import React from 'react';
import { Text, TextInput, TouchableHighlight, View, FlatList } from 'react-native';

export default class App extends React.Component {
  constructor(props){
    super(props);
    var movies = [{key:"Star Wars"},{key:"Back to the Future"},
                  {key:"The Matrix"},{key:"Inception"},{key:"Interstellar"}]
    this.state = {movies: movies, valueTextInput: ""}
  }
  _addMovieFromTextInput(){
    if((this.state.valueTextInput.trim() != "")&&(this.state.movies.map(function(obj){
    return obj.key}).indexOf(this.state.valueTextInput)==-1)){
      this._addMovie({key: this.state.valueTextInput});
    }
    this.setState({valueTextInput: ""});
  }
  [...]
  render() {[...]}
}
```

Ejemplo: Lista dinámica con FlatList

```
import React from 'react';
import { Text, TextInput, TouchableHighlight, View, FlatList } from 'react-native';

export default class App extends React.Component {
  [...]
  _addMovie(movie){
    var movies = Object.assign([], this.state.movies);
    movies.push(movie);
    this.setState({movies: movies});
  }
  _removeMovie(movie){
    var movies = Object.assign([], this.state.movies);
    for(let i=0; i<movies.length; i++){
      if(movies[i].key === movie.key){
        movies.splice(i,1);
      }
    }
    this.setState({movies: movies});
  }
  [...]
  render() {[...]}
}
```

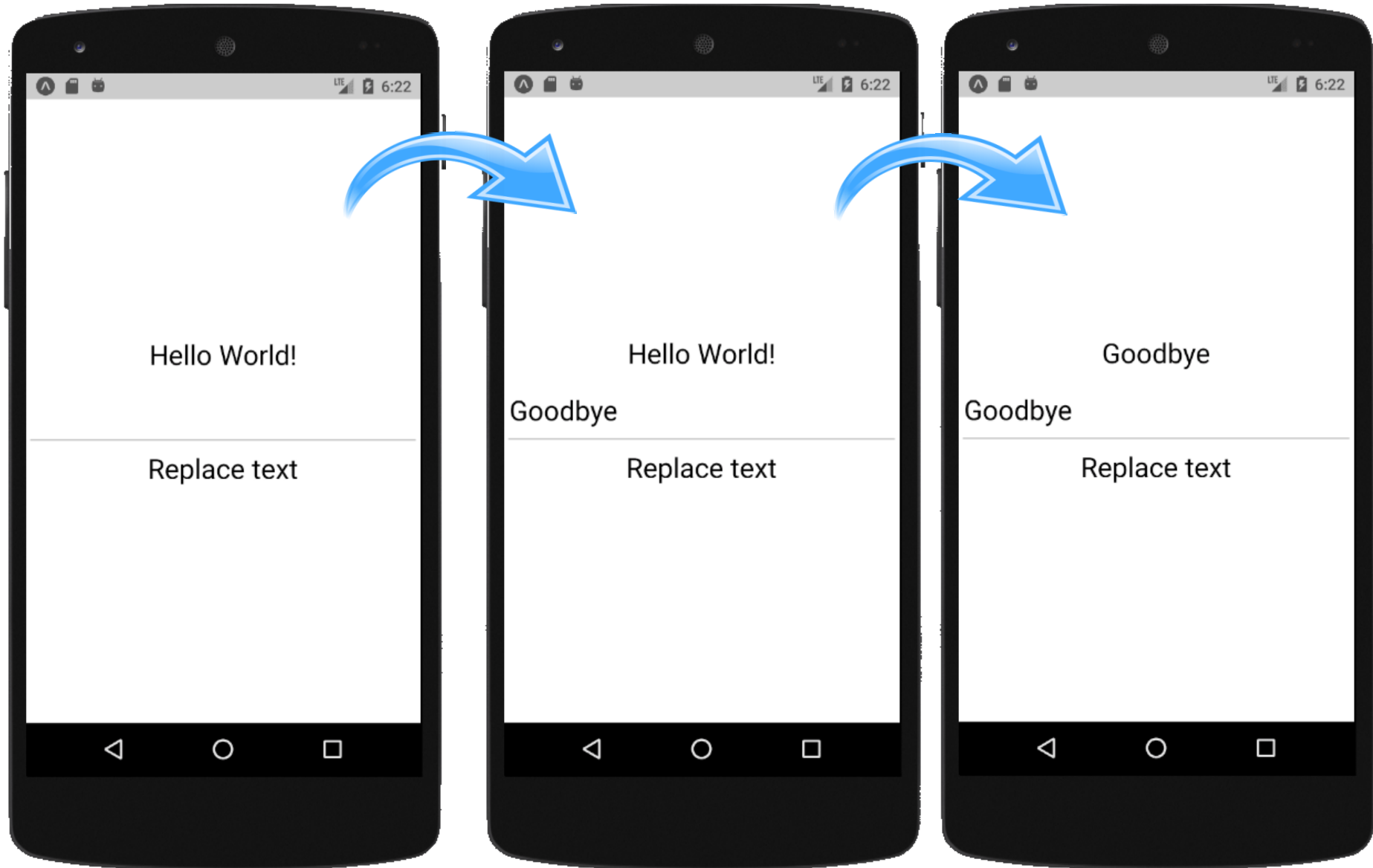
Ejemplo: Lista dinámica con FlatList

```
import React from 'react';
import { Text, TextInput, TouchableHighlight, View, FlatList } from 'react-native';
export default class App extends React.Component {
  [...]
  _renderItem({item}){
    return (
      <View style={{flexDirection: 'row'}}>
        <Text style={{fontSize: 30}}>{item.key}</Text>
        <TouchableHighlight onPress={() => this._removeMovie(item)}>
          <Text style={{fontSize:30, color:'red'}}> [X] </Text>
        </TouchableHighlight>
      </View>
    )
  }
  [...]
  render() {[...]}
}
```


Ejercicio React Native

1. Instalar React Native
2. Crear y ejecutar la aplicación Hello World inicial
3. Modificar la aplicación Hello World como se indica en las transparencias 'Modificación de Hello World'
4. Realizar los siguientes cambios a la aplicación:
 - Añadir una caja de texto editable ***TextInput***
 - Añadir un botón ***TouchableHighlight*** que al pulsarlo reemplace el contenido del elemento ***Text*** que muestra inicialmente "Hello World!" por el texto escrito en el ***TextInput***
 - Posicionar todos los elementos usando **Flexbox** de modo que aparezcan dispuestos verticalmente de forma consecutiva y centrados en la pantalla (horizontal y verticalmente)

Ejercicio React Native



¿Preguntas?

Aldo Gordillo

agordillo@dit.upm.es