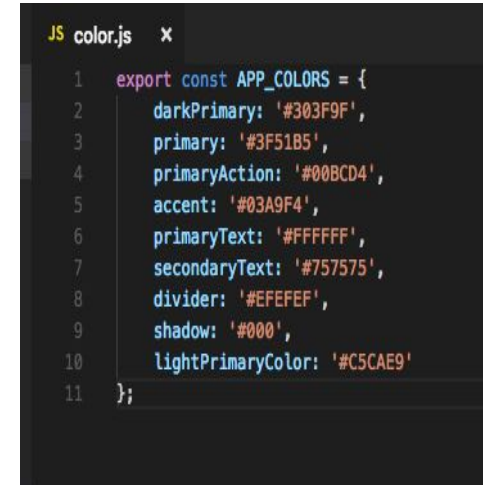
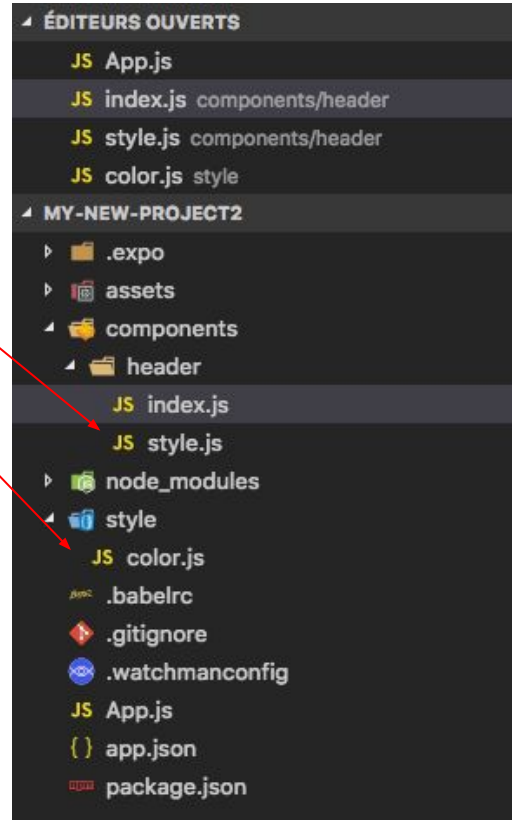


# Le style

On crée un fichier style.js qui gère le style de notre composant “header”.

Le fichier color est un fichier qui “export” ces couleurs.



On encapsule la balise `<Text>` avec la balise `<View>`, la balise `<View>` devient le conteneur de `<Text>` cela veut dire que l'on a le choix maintenant de stylé le conteneur ou uniquement le text.

```
JS index.js x
1  import React from 'react';
2  import { Text, View } from 'react-native';
3
4  const Header = ({content}) => {
5
6      return <View>
7              <Text>{content}</Text>
8              </View>
9
10 };
11 export default Header;
```

Dans le fichier "style.js" c'est là que l'on va créer notre feuille de style pour ensuite le lier à la vue que l'on désire.

L'import {StyleSheet} donne accès au composant de react-native concerné.

L'import APP\_COLORS est le fichier où l'on a répertorié toutes nos couleurs, APP\_COLORS est l'objet qui contient toutes les clés pour accéder aux propriétés de nos couleurs.

Note : Le composant header a son propre fichier de style.js situé dans le même fichier.

```
JS index.js  JS style.js  x  JS color.js
1  import {StyleSheet} from 'react-native';
2  import {APP_COLORS} from '../styles/color';
3
4  export const style = StyleSheet.create({
5    subHeader:{
6      backgroundColor: APP_COLORS.darkPrimary,
7      height: 30
8    }
9  });
```

```
components
└─ header
   JS index.js
   JS style.js
node_modules
style
```

```
JS color.js  x
1  export const APP_COLORS = {
2    darkPrimary: '#303F9F',
3    primary: '#3F51B5',
4    primaryAction: '#00BCD4',
5    accent: '#03A9F4',
6    primaryText: 'FFFFFF',
7    secondaryText: '#757575',
8    divider: 'EFEFEF',
9    shadow: '000',
10   lightPrimaryColor: 'C5CAE9'
11  };
```

On importe ensuite la variable style du fichier style.js dans le composant concerné.

On utilise le mot clé 'style' définit la propriété du composant qui gère le style.


La propriété 'style' reçoit l'objet style.

Note : On fournit a la propriété 'style' l'objet {style.subHeader} car un objet est définis par des clé (style) et valeur(subheader), d'ailleurs (subheader) est également une clé.

```
JS index.js x JS style.js JS color.js
1 import React from 'react';
2 import { Text, View } from 'react-native';
3 import { style } from './style';
4
5 const Header = ({content}) => {
6
7   return <View >
8     <View style={style.subHeader}/>
9     <View><Text>{content}</Text></View>
10    </View>
11
12  };
13 export default Header;
```

```
JS index.js JS style.js x JS color.js
1 import {StyleSheet} from 'react-native';
2 import {APP_COLORS} from '../../styles/color';
3
4 export const style = StyleSheet.create({
5   subheader:{
6     backgroundColor: APP_COLORS.darkPrimary,
7     height: 30
8   }
9 });
```

On peut stylé d'autre vue en ajoutant un objet dans l'objet style{ }.



```
JS index.js JS style.js x JS color.js
1 import {StyleSheet} from 'react-native';
2 import {APP_COLORS} from '../styles/color';
3
4 export const style = StyleSheet.create({
5   subHeader:{
6     backgroundColor: APP_COLORS.darkPrimary,
7     height: 30
8   },
9   header:{
10     backgroundColor: APP_COLORS.primary,
11     height: 150
12   }
13 });
```

```
JS index.js x JS style.js JS color.js
1 import React from 'react';
2 import { Text, View } from 'react-native';
3 import { style } from './style';
4
5 const Header = ({content}) => {
6
7   return (
8     <View style={style.subHeader}>
9       <View style={style.header}>
10         <Text>{content}</Text></View>
11       </View>
12   );
13 };
14 export default Header;
```

# Le Positionnement

Le fait d'effectuer des modifications de positionnement sur la vue 'header' on modifie le positionnement qui se situe à l'intérieur de la vue dans l'exemple, ci dessous on agit sur le style de positionnement de 'header' et on a un tag <text> a l'interieur que l'on positionne de tel façon qu'il soit centrer.

```
JS index.js x JS style.js JS color.js
1 import React from 'react';
2 import { Text, View } from 'react-native';
3 import { style } from './style';
4
5 const Header = ({content}) => {
6
7   return
8     <View >
9       <View style={style.subHeader}/>
10      <View style={style.header}>
11        <Text>{content}</Text></View>
12      </View>
13    };
14 export default Header;
```

```
JS index.js JS style.js x JS color.js
1 import {StyleSheet} from 'react-native';
2 import {APP_COLORS} from '../styles/color';
3
4 export const style = StyleSheet.create({
5   subHeader:{
6     backgroundColor: APP_COLORS.darkPrimary,
7     height: 30
8   },
9   header:{
10     backgroundColor: APP_COLORS.primary,
11     height: 150,
12     justifyContent: 'center',
13     alignItems: 'center',
14   },
15   text:{
16     color: APP_COLORS.primaryText,
17     fontSize: 30
18   }
19 });
```

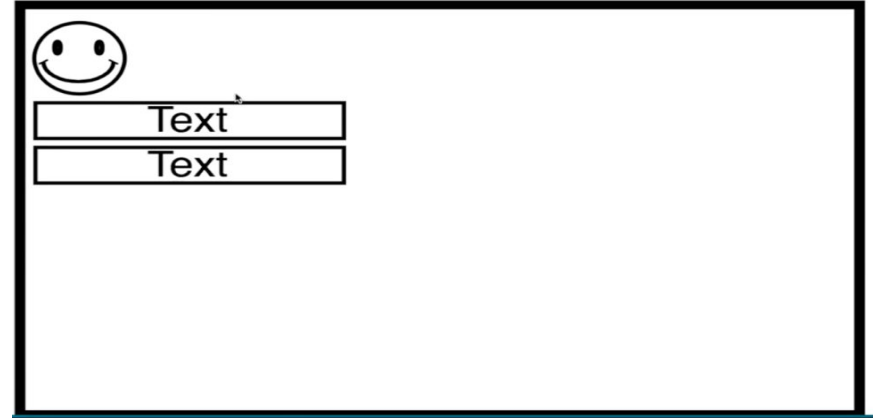


Par défaut le flex est paramétré comme cela :

`flex-direction: 'column'`



Comportement par défaut

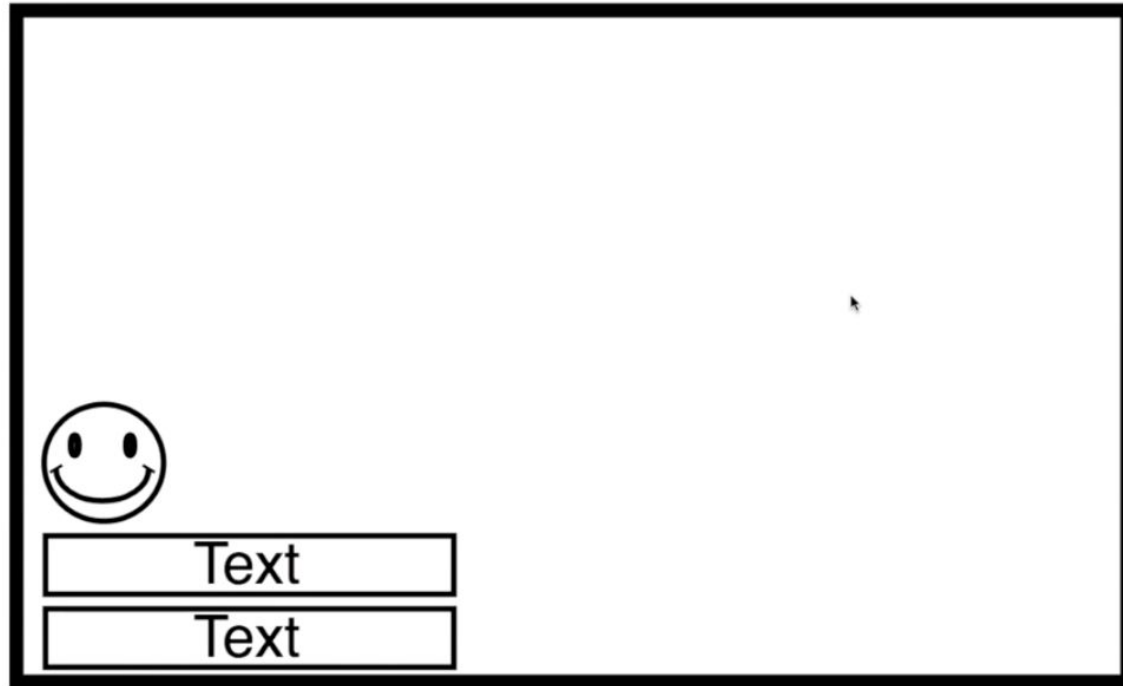


Si on aurait mis 'row' a la place de column il se serait mis les uns à côté des autres.



Le 'justifyContent' permet de travailler sur la **colonne** dans le cas d'un 'flex-direction = **column**',  
ou  
Le 'justifyContent' permet de travailler sur la **ligne** dans le cas d'un 'flex-direction = **row**,

justifyContent: 'flex-end'



justifyContent: 'center'



Text

Text

justifyContent: 'space-between'



Text

Text

justifyContent: 'space-around'



Text

Text

flex-direction: 'row'



Text

Text

flex-direction: 'row'  
justifyContent: 'space-around'



Text

Text

‘alignItems’ permet de s’occuper de l’axe dont  
‘justifyContent’ ne s’occupe pas.

exemple:

flex-direction : column,  
justifyContent : center,  
alignItems : center

Cela va centrer les éléments.

# Exemple:

```
JS index.js  JS style.js  x  JS color.js
1  import {StyleSheet} from 'react-native';
2  import {APP_COLORS} from '../..../styles/color';
3
4  export const style = StyleSheet.create({
5    subHeader:{
6      backgroundColor: APP_COLORS.darkPrimary,
7      height: 30
8    },
9    header:{
10     backgroundColor: APP_COLORS.primary,
11     height: 150,
12     justifyContent: 'center',
13     alignItems: 'center',
14     shadowColor: APP_COLORS.shadow,
15     shadowOpacity: 0.2,
16     shadowOffset: {height:10},
17   },
18   text:{
19     color: APP_COLORS.primaryText,
20     fontSize: 30
21   }
22 });
```

