# Annexe 3 – Les variables d'état "state"

Le concept de state est très important pour les composants React et React Native. Ces variables représentent la mémoire de la composante.

En effet, les variables locales à un composant ne sont pas conservées entre les réaffichages des composants, suite à un clic ou à un autre événement. Parallèlement, le changement de valeur d'une variable ne force pas le réaffichage du composant.

## Voir contre-exemple en classe : chefs d’œuvre de la musique classique

On peut utiliser les "state" de deux manières :

## Dans une classe :

Toujours appeler la variable state

Class App extends react component

{

State = {valeur : 1, val2 : 3} <- variable d’état : conserve leurs valeurs lors des réaffichages

gererButton = () => this.setState({ valeur :this.state.valeur+1, val2: this.state.val2\*3})

Render()

{

Return( <View>

<Button title= ‘’click’’ onPress= {this.gererButton} />

<Text> {this.state.val2} </Text>

</View>;

)

}

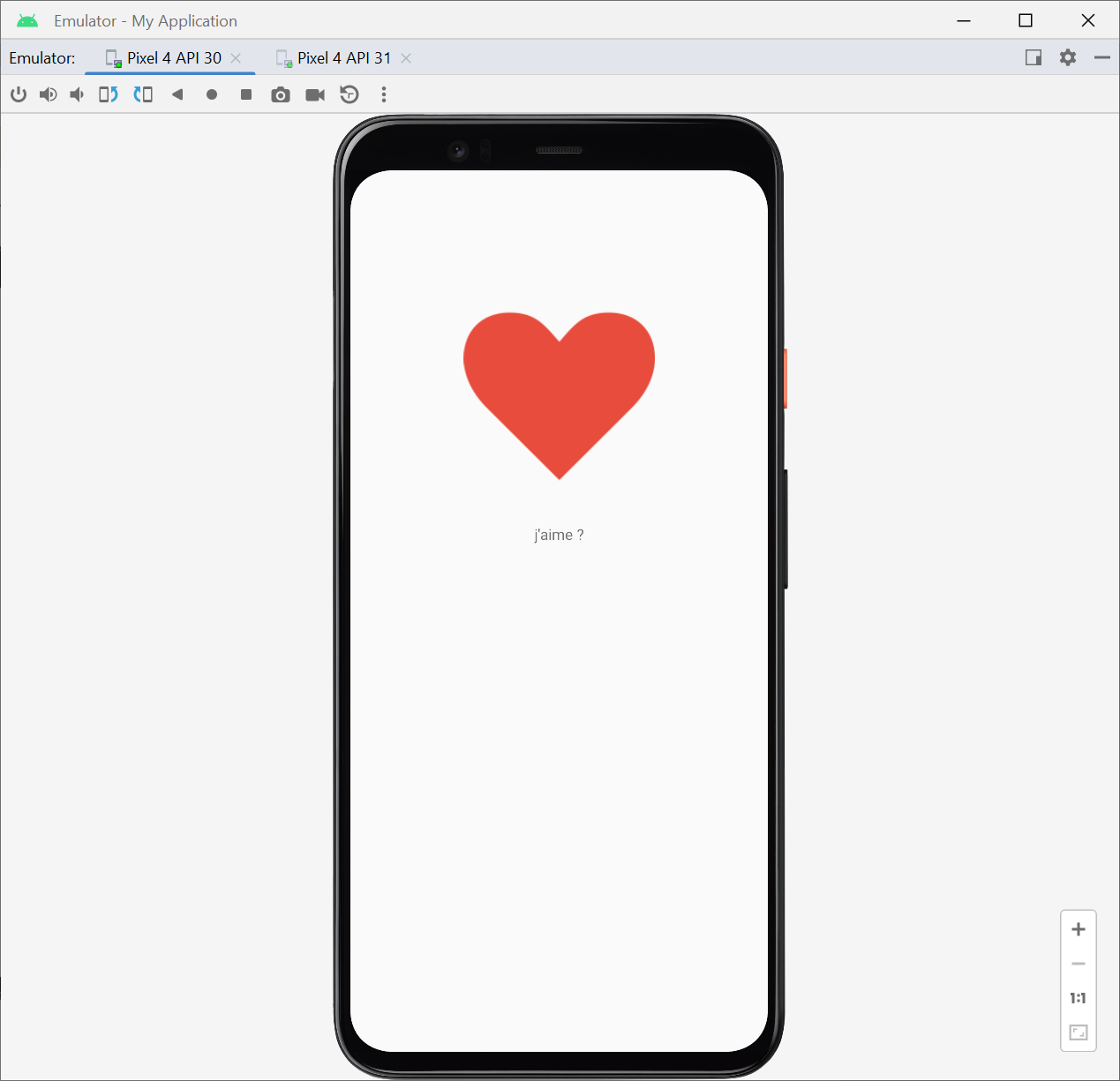
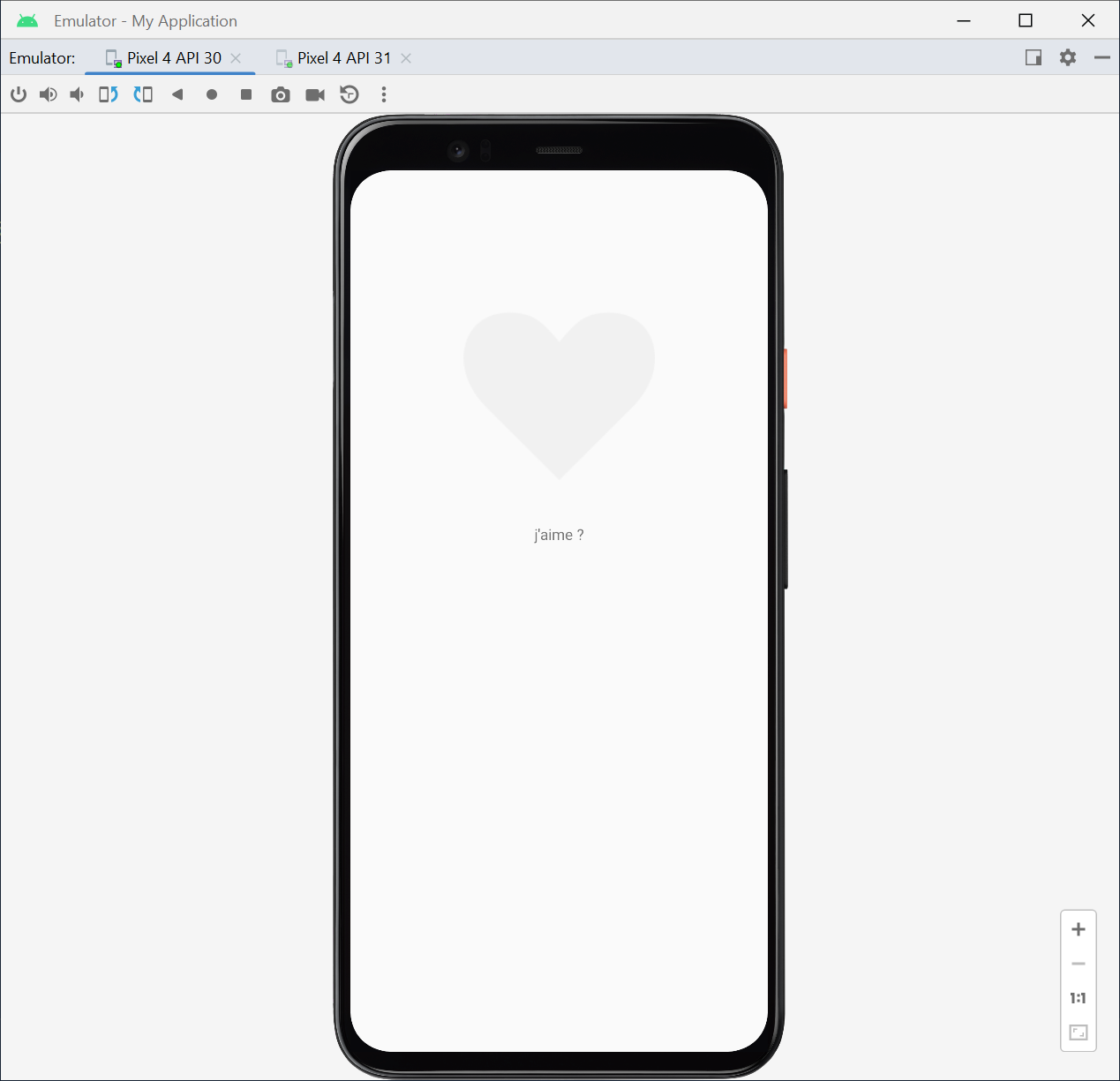
}

## PRIVILÉGIÉ PAR RAPPORT AUX CLASSES

## Dans un composant fonctionnel, utiliser plutôt le hook useState :

* En React, toutes les méthodes qui commencent par use sont surnommées des « hooks »
* Ce sont des fonctions spéciales qui sont disponibles quand React est en rafraîchissement : au départ et lorsque le hook change de valeur
* Par convention, les hooks sont déclarés au début de la fonction / composant

## Exercice : « J’aime » ou pas…



Le cœur devient rouge lorsqu'on clique dessus et redevient transparent lorsqu'on reclique

1. Développer un projet avec un composant fonctionnel composé d'un composant Image et d'un composant Text. L'image est fournie sur LÉA. Vous pouvez associer un StyleSheet à votre image; la propriété ( style props ) tintColor #f1f1f1 permettra une apparence translucide.
2. Ce composant aura combien d'états visuels ? De quel type sera donc la variable d'état ? Utiliser un useState avec une valeur initiale appropriée.

\*\*\* On doit importer useState de la librairie React \*\*\*

1. On n’a visiblement pas besoin de composant Button; vous pouvez à la place utiliser le composant TouchableOpacity pour englober votre image.
   * Quand on presse sur le TouchableOpacity, l'opacité de la composante diminue
   * Le TouchableOpacity ne peut contenir qu'un seul enfant
   * On peut lui associer un événement onPress où vous pourrez modifier la variable d'état
   * On peut modifier la couleur du cœur avec la variable d'état en y associant une nouvelle règle de l'objet StyleSheet, avec un opérateur ternaire :

Ex. :  const styleCoeur = aime ? styles.aime : null; où aime est la variable d'état