REACT NATIVE MUSIC DEMO

Perry Wu

2022/1/12

Establish a lib project.

Reference base project

- https://www.ideamotive.co/blog/how-to-create-your-first-react-native-npm-package-in-typescript
- https://www.npmjs.com/package/react-native-module-template/v/2.3.2

Build step

- # git clone https://github.com/demchenkoalex/react-native-module-template.git
- # mv react-native-module-template demo && cd demo
- # node rename.js

Rename as you want (optional)

```
perry@PerrydeMacBook-Pro library % mv react-native-module-template demo && cd demo
perry@PerrydeMacBook-Pro demo % node rename.js
Enter library name (use kebab-case) (default react-native-module-template): MusicDemo
Enter library short name (used to name Swift and Kotlin classes, use PascalCase) (default RNModuleTemplate): MD
Enter library homepage (default https://github.com/demchenkoalex/react-native-module-template#readme):
Enter library git url (default https://github.com/demchenkoalex/react-native-module-template.git):
Enter author name (default Alex Demchenko):
Enter author email (default alexdemchenko@yahoo.com):
perry@PerrydeMacBook-Pro demo %
```

Build step

- # rm rename.js && yarn install
- # cd example && yarn install

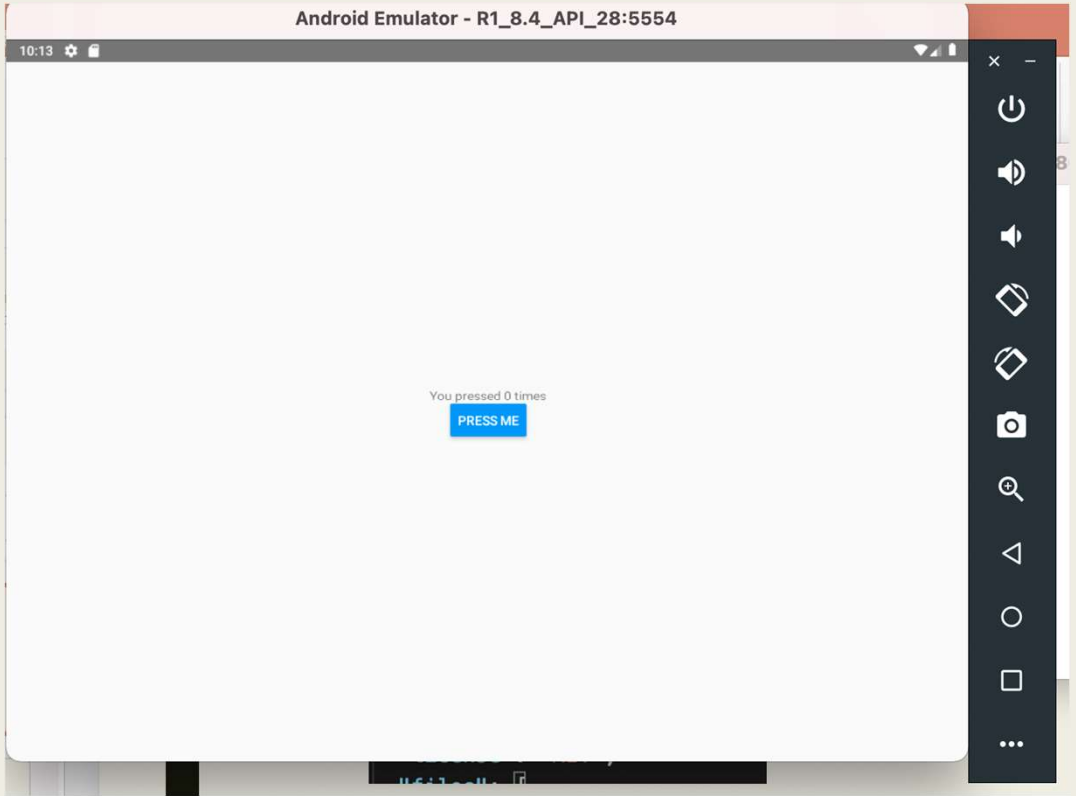
```
perry@PerrydeMacBook-Pro demo % rm rename.js
perry@PerrydeMacBook-Pro demo % yarn install
yarn install v1.22.17
[1/4] 🔍  Resolving packages...
[2/4] 🚚  Fetching packages...
[3/4] 🔗  Linking dependencies...
warning " > @react-native-community/eslint-config@3.0.1" has unmet peer dependency "prettier@>=2".
warning "react-native > react-native-codegen > jscodeshift@0.11.0" has unmet peer dependency "@babel/preset-env@^7.1.6".
[4/4] 🔨  Building fresh packages...
$ yarn compile
yarn run v1.22.17
$ rm -rf lib && tsc -p .
✨  Done in 1.49s.
✨  Done in 7.26s.
perry@PerrydeMacBook-Pro demo % cd example && yarn install
yarn install v1.22.17
[1/4] 🔍  Resolving packages...
[2/4] 🚚  Fetching packages...
[3/4] 🔗  Linking dependencies...
warning "react-native > react-native-codegen > jscodeshift@0.11.0" has unmet peer dependency "@babel/preset-env@^7.1.6".
warning " > @react-native-community/eslint-config@3.0.1" has unmet peer dependency "prettier@>=2".
[4/4] 🔨  Building fresh packages...
✨  Done in 5.92s.
perry@PerrydeMacBook-Pro example %
```

Build step

- In the root folder in package.json change main to “src/index.tsx”

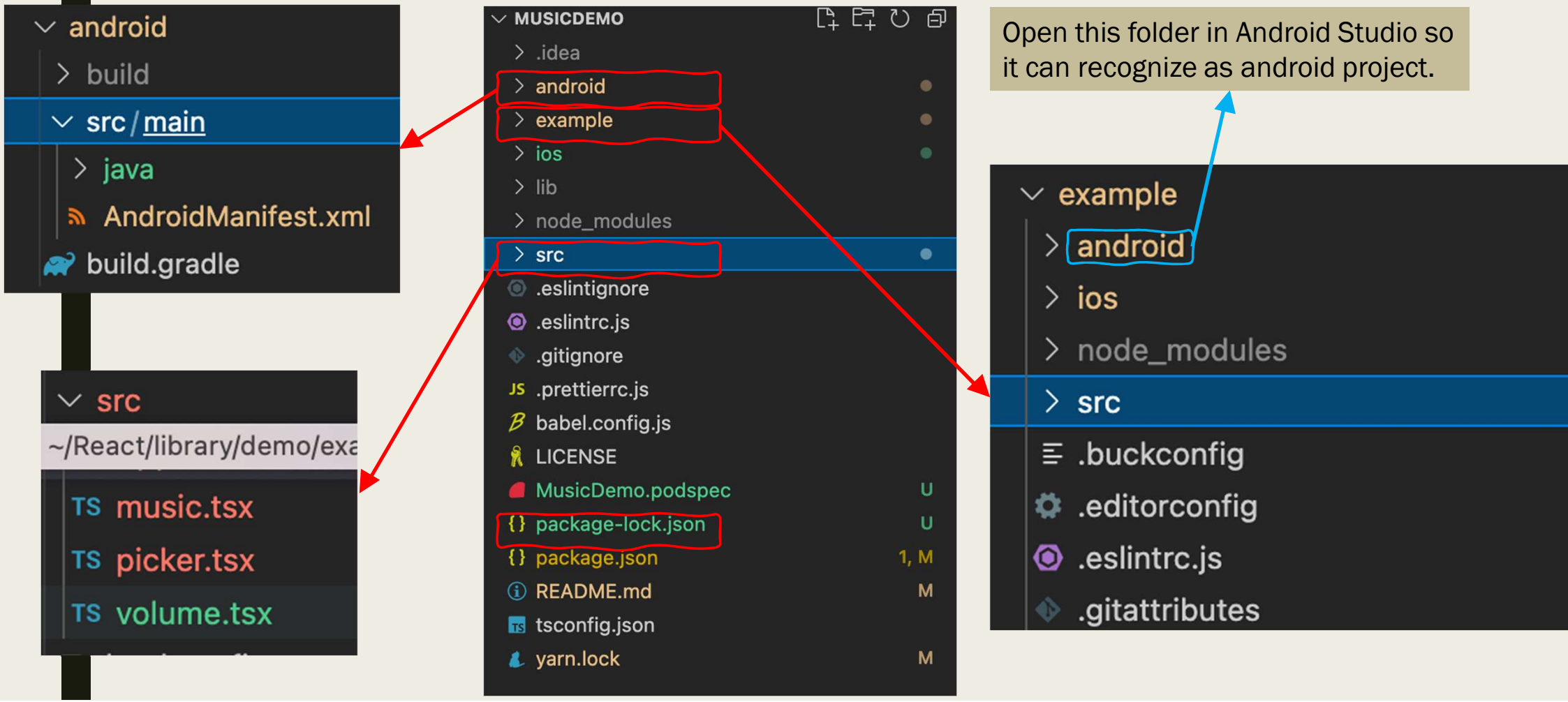
```
{
  "name": "MusicDemo",
  "version": "1.0.0",
  "description": "",
  "homepage": "https://github.c
  "main": "src/index.tsx",
  "types": "lib/index.d.ts",
  "author": "Alex Demchenko <al
  "license": "MIT",
```

.tsx is typescript with JSX.

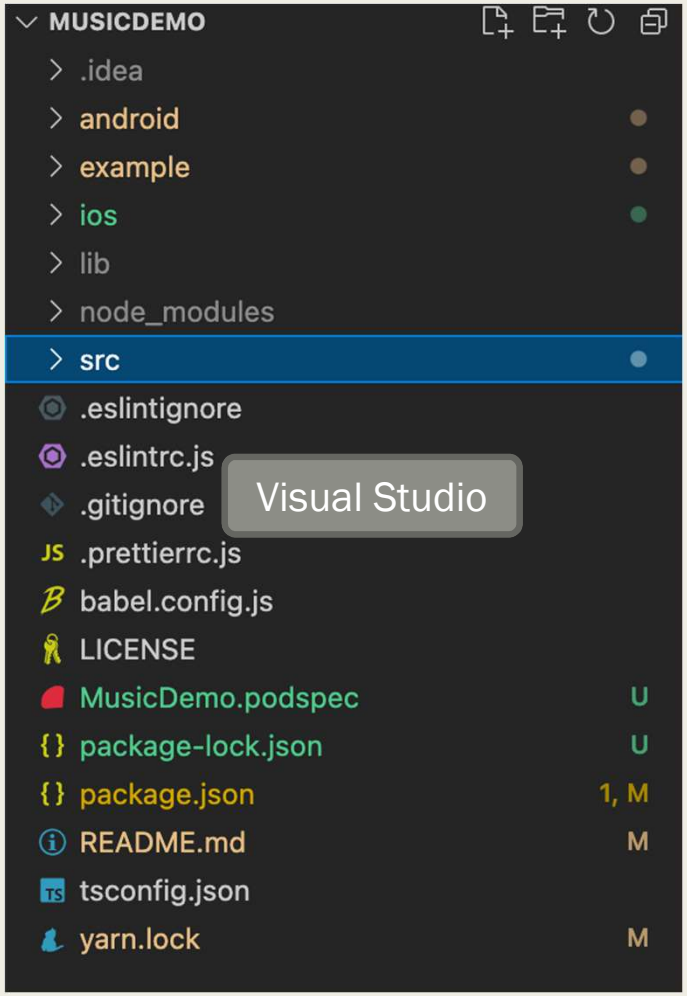


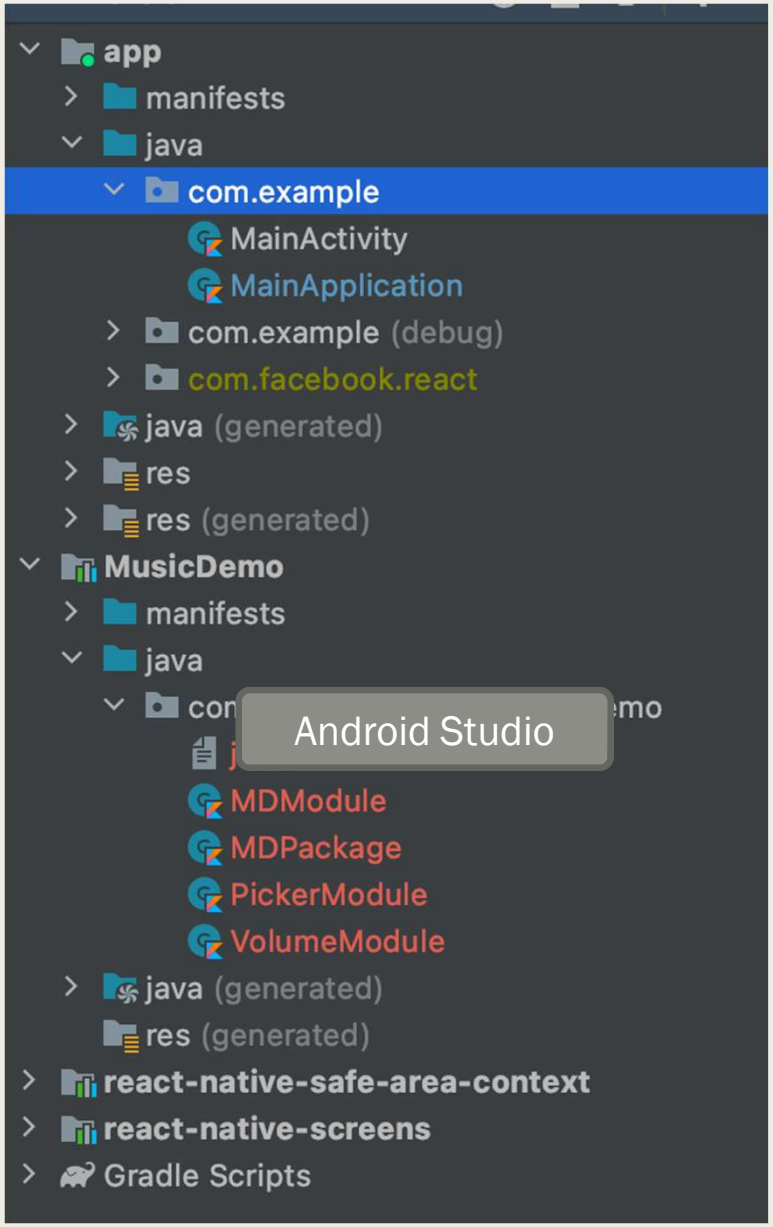
- # yarn start
- # npx react-native run-android

File Structure



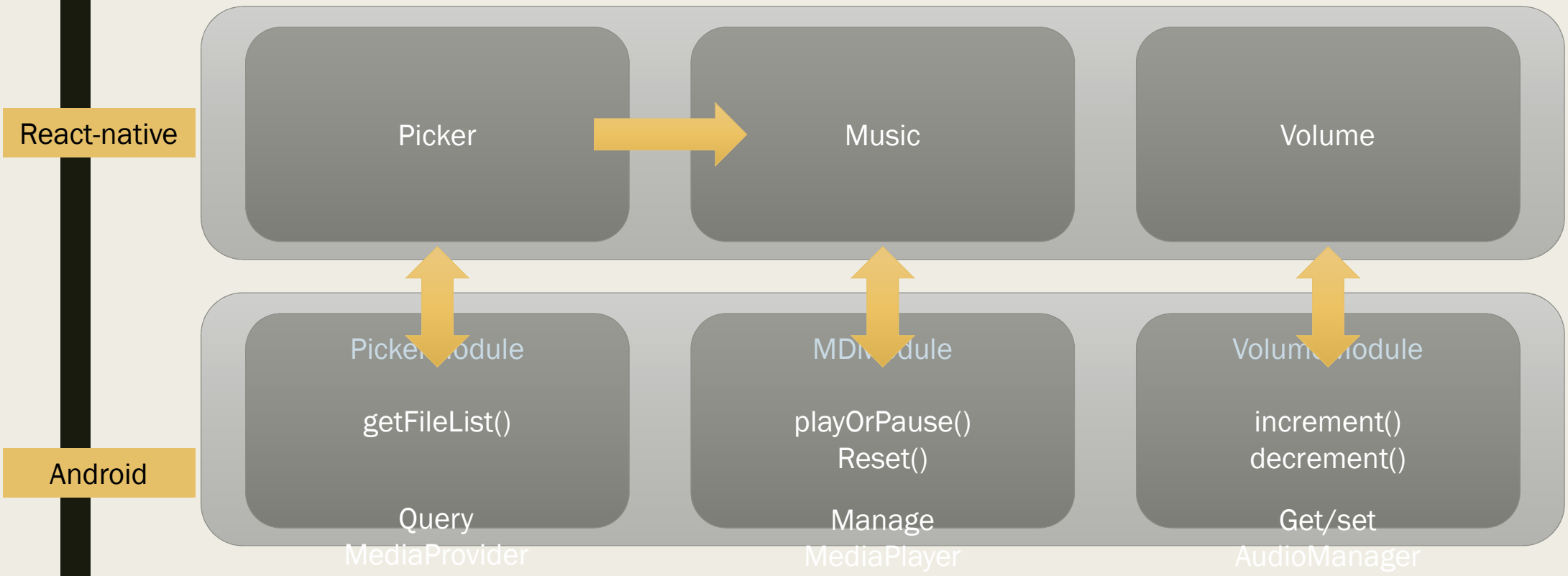
File Structure

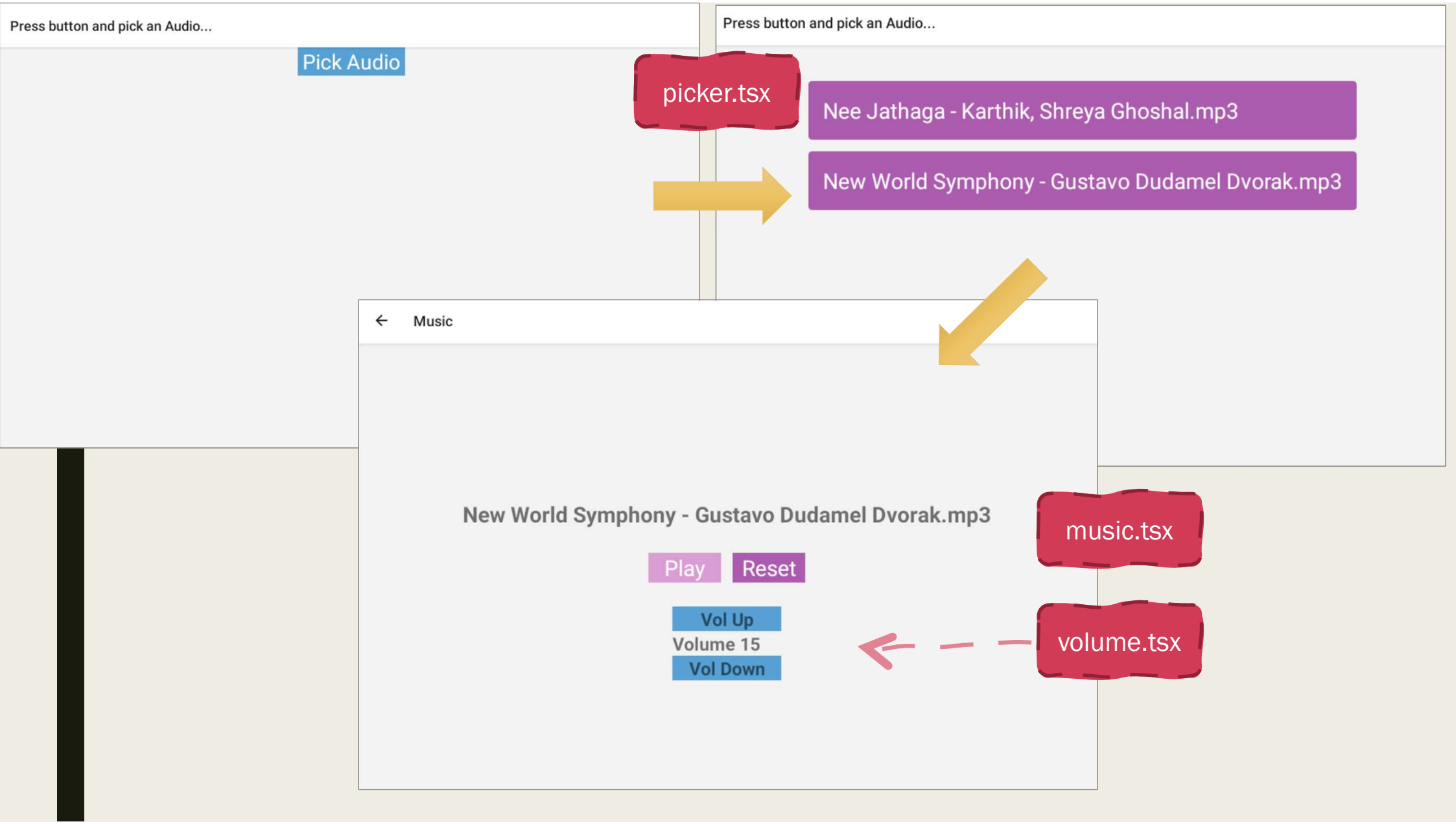




Implementation

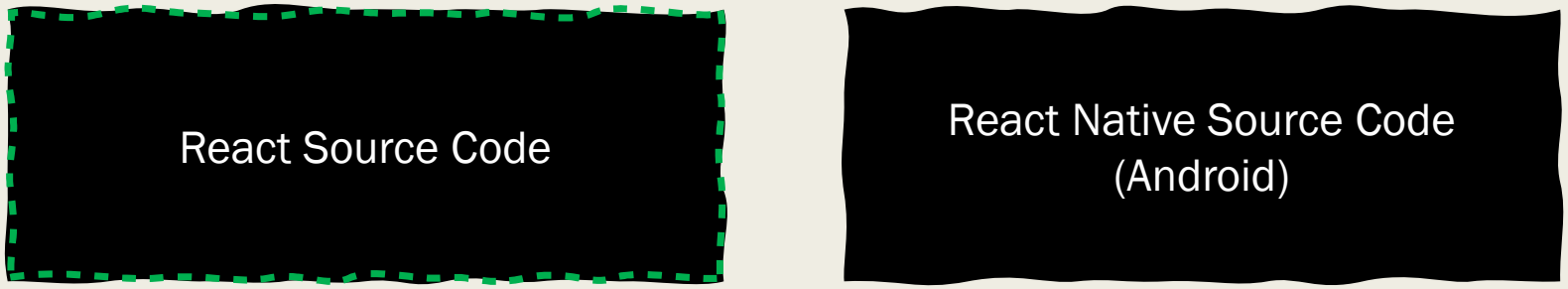
Architecture




Press button and pick an Audio...
Pick Audio
picker.tsx
Press button and pick an Audio...
Nee Jathaga - Karthik, Shreya Ghoshal.mp3
New World Symphony - Gustavo Dudamel Dvorak.mp3
Music
New World Symphony - Gustavo Dudamel Dvorak.mp3
Play
Reset
Vol Up
Volume 15
Vol Down
music.tsx
volume.tsx

Call modules

- At following slide, react code is surrounded by dotted green frame.



React - App.js

```
import { Platform, StyleSheet, Text, View, Button, NativeModules, NativeEventEmitter } from 'react-native';
```

```
const { MDModule , PickerModule} = NativeModules;
```

```
render() {
  return (
    <View style={styles.container}>
      <Button onPress={ () => this.play} title={this.state.title}
      <Button onPress={ () => this.reset} title='Reset' />
      <Volume/>
      <Picker />
    </View>
  );
}
```

```
play() {
  MDModule.playOrPause()
  .then( (title: string) => this.setState({tit
  .catch( (error: any) => console.error(error)
}

reset() {
  MDModule.reset()
}
```

Android Native – create module (1/2)

```
class MDModule(private val reactContext: ReactApplicationContext) :
    ReactContextBaseJavaModule(reactContext),
    LifecycleEventListener {

    private var mMediaPlayer = MediaPlayer()
    private var initialized = false

   init {
       reactContext.addLifecycleEventListener( listener: this);
   }

    override fun getName() = "MDModule"

    override fun getConstants(): MutableMap<String, Any> {
        return hashMapOf("initial" to "Play")
    }
```

Android Native – create module (2/2)

- In order to expose a method from a native Java module to JS, just write a method and add @ReactMethod annotation on top of it.

```
@ReactMethod
fun playOrPause(promise : Promise) {
    try {
        if (!initialized) prepare()

        if (mMediaPlayer.isPlaying) {
            mMediaPlayer.pause()
            promise.resolve( value: "Play")
        } else {
            mMediaPlayer.start()
            promise.resolve( value: "Pause")
        }
    } catch (e : Exception) {
        promise.reject( code: "E_ERROR", e)
    }

}
```

Android Native – register module

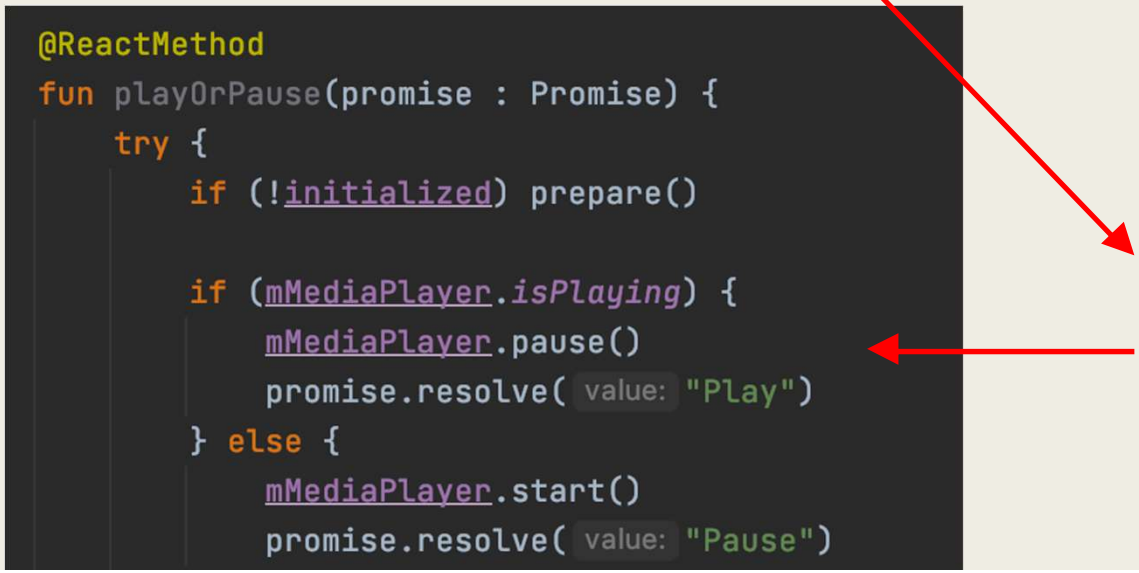
```
class MDPackage : ReactPackage {

    override fun createViewManagers(reactContext: ReactApplicationContext):
            MutableList<ViewManager<*, *>> {
        return mutableListOf()
    }

    override fun createNativeModules(reactContext: ReactApplicationContext):
            MutableList<NativeModule> {
        return mutableListOf(
            MDModule(reactContext),
            VolumeModule(reactContext),
            PickerModule(reactContext))
    }
}
```

Export Constants

```
class MDModule(private val reactContext: ReactApplicationContext) :
    ReactContextBaseJavaModule(reactContext),
    LifecycleEventListener {

    private var mMediaPlayer = MediaPlayer()
    private var initialized = false

   init {
       reactContext.addLifecycleEventListener
   }

    override fun getName() = "MDModule"

    override fun getConstants(): MutableMap<String, Any> {
        return hashMapOf("initial" to "Play")
    }
```

```
export class Music extends Component<Props, State> {
  constructor(props: Props) {
    super(props);
    this.state = {
      title: MDModule.initial
    };
  }
```

Promise



- Similar to callbacks, a native module method can either reject or resolve a promise (but not both) and can do so at most once.

EventEmitter (1/2)

```
private void sendEvent(ReactContext reactContext, String eventName, @Nullable Writab
    reactContext.getJSModule(DeviceEventManagerModule.RCTDeviceEventEmitter.class)
            .emit(eventName, params);
}
```

```
@ReactMethod
fun reset() {
    mMediaPlayer.reset()
    initialized = false
    val params = Arguments.createMap()
    params.putString("title", "Replay")
    sendEvent(this.reactApplicationContext,  eventName: "onComplete", params)
}

private fun prepare() {
    val filePath : String = reactContext.getExternalFilesDir(Environment.DIRECTORY_DOWNLOADS)?.absolutePath + "/head.m4a"
    println("MDModule + $filePath")
    mMediaPlayer.setDataSource(filePath)
    mMediaPlayer.setOnCompletionListener {  it: MediaPlayer!
        println("OnCompletionListener")
        reset()
    }
    mMediaPlayer.prepare()
    initialized = true
}
```

EventEmitter (2/2)

Native modules can signal events to JavaScript without being invoked directly.

```
const completeEvents = new NativeEventEmitter(MDModule);
const onComplete = 'onComplete'
```

```
UNSAFE_componentWillMount() {
  completeEvents.addListener(onComplete, ({ title }) => {
    this.setState({ title });
  });
}

componentWillUnmount() {
  completeEvents.removeAllListeners(onComplete);
}
```

Listen to Android LifeCycle

```
reactContext.addLifecycleEventListener(this);
```

```
override fun onHostResume() {
    //TODO("Not yet implemented")
}

override fun onHostPause() {
    //TODO("Not yet implemented")
}

override fun onHostDestroy() {
    println("onHostDestroy")
    mMediaPlayer.release()
}
```

Argument Type

For argument types not listed above, you will need to handle the conversion within the native method yourself.

JAVA	JAVASCRIPT
Boolean	?boolean
boolean	boolean
Double	?number
double	number
String	string
Callback	Function
ReadableMap	Object
ReadableArray	Array

Passing Array from Android to UI (1/3)

- Target : Pass Audio[] to UI

WritableNativeArray()

- - WritableNativeMap()

...

- - WritableNativeMap()

```
data class Audio(
    val uri: Uri,
    val name: String,
    val duration: Int,
    val size: Int
) {
```

```
val map = WritableNativeArray()

query?.use{...}

promise.resolve(map)
```

```
val content: WritableMap = WritableNativeMap()
content.putString("uri", audio.uri.toString())
content.putString("name", audio.name)
content.putInt("duration", audio.duration)
content.putInt("size", audio.size)

map.pushMap(content)
```

Passing Array from Android to UI (2/3)

WritableNativeArray()

- - WritableNativeMap()

...

- - WritableNativeMap()

```
data class Audio(
    val uri: Uri,
    val name: String,
    val duration: Int,
    val size: Int
) {
```


```
{
    "size":10805746,
    "duration":265561,
    "name":"001 - 小幸運.mp3",
    "uri":"content://media/external/video/media/33"
},
{
    "size":8797604,
    "duration":217757,
    "name":"002 - 多遠都要在一起.mp3",
    "uri":"content://media/external/video/media/34"
},
{
    "size":5604537,
    "duration":232385,
    "name":"003 - 愛不需要裝乖 (feat. 王詩安 Diana).mp3",
    "uri":"content://media/external/video/media/35"
},
```

Passing Array from Android to UI (3/3)

WritableNativeArray()

- - WritableNativeMap()

  ...

- - WritableNativeMap()

```
data class Audio(
    val uri: Uri,
    val name: String,
    val duration: Int,
    val size: Int
) {
```

FlatList

```
import { Button, NativeModules, StyleSheet, View, FlatList } from 'react-native'
```

```
export const Picker = () => {
  const navigation = useNavigation();
  const [response, setResponse] = useState<AudioInfo[]>([]);

  const view  = () => {
    PickerModule.getFileList()
    .then( (data: AudioInfo[]) => setResponse(data))
    .catch( (error: any) => console.error(error))
```

```
<FlatList
  data={response}
  extraData={response}
  renderItem={({ item }) => (
    <Button onPress={ () => navigateToMusic(item)} title={`${item.name}`}/>
  )}
/>
```

FlatList

```
import { Button, NativeModules, StyleSheet, View, FlatList } from 'react-native'
```

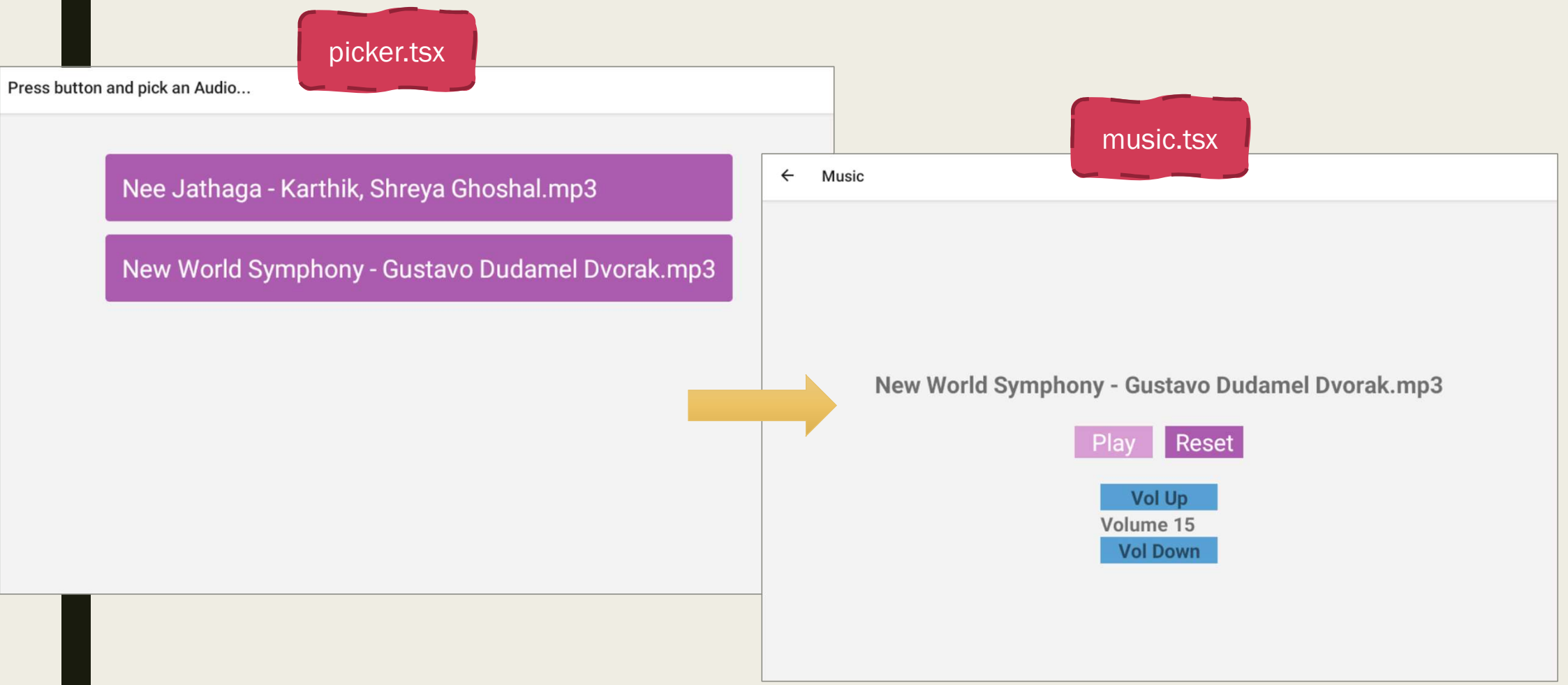
```
export const Picker = () => {
  const navigation = useNavigation();
  const [response, setResponse] = useState<AudioInfo[]>([]);

  const view  = () => {
    PickerModule.getFileList()
    .then( (data: AudioInfo[]) => setResponse(data))
    .catch( (error: any) => console.error(error))
```

```
<FlatList
  data={response}
  extraData={response}
  renderItem={({ item }) => (
    <Button onPress={ () => navigateToMusic(item)} title={`${item.name}`}/>
  )}
/>
```

Navigation (1/3)

```
# npm install @react-navigation/native @react-navigation/native-stack
# npm install react-native-screens react-native-safe-area-context
```

Navigation


picker.tsx
Press button and pick an Audio...
Nee Jathaga - Karthik, Shreya Ghoshal.mp3
New World Symphony - Gustavo Dudamel Dvorak.mp3
music.tsx
Music
New World Symphony - Gustavo Dudamel Dvorak.mp3
Play
Reset
Vol Up
Volume 15
Vol Down

Navigation (2/3)

```
function navigateToMusic(item : AudioInfo) {
  setSelectedPath(item.path)
  navigation.navigate("Music",
                      { name : item.name,
                        path : item.path });
}
```

Navigation (3/3) - Route

```
import { useRoute } from '@react-navigation/native';
```

```
const Music = () => {
  const route = useRoute();
```

```
const play = () => {
  console.log("play...")
  console.log(`${route.params.path}`)

  MDModule.playOrPause(route.params.path)
  .then( (action: string) => setAction(action))
  .catch( (error: any) => console.error(error));
}
```

Reference

- https://reactnative.dev/docs/native-modules-android
- https://github.com/mitsuruog/react-native-call-native-module-sample
- https://reactnavigation.org/docs/hello-react-navigation