Лекція-практикум 4

Використання Image Picker

У цій лекції ви дізнаєтесь, як скористатися **Expo Image Picker**. React Native надає стандартні будівельні блоки (`<View>`, `<Text>`, `<Pressable>`), але вибір зображення з медіатеки пристрою потребує окремої бібліотеки. Ми будемо використовувати **`expo-image-picker`** з Expo SDK.

`expo-image-picker` відкриває системний інтерфейс для вибору зображень і відео з медіатеки (а також може працювати з камерою).

1) Встановити `expo-image-picker`

У терміналі виконайте:

```
- npx expo install expo-image-picker
```

Після інсталяції, якщо сервер розробки працював, зупиніть його (`Ctrl+C`) і перезапустіть: `npx expo start`.

2) Вибрати зображення з медіатеки пристрою

`expo-image-picker` надає метод `launchImageLibraryAsync()` для показу системного UI вибору медіа. Додамо функцію до екрану, щоб відкривати медіатеку й обробляти результат.

`app/(tabs)/index.tsx` (фрагмент):

```tsx
// ...rest of the import statements remain unchanged
import * as ImagePicker from 'expo-image-picker';

export default function Index() {
  const pickImageAsync = async () => {
    let result = await ImagePicker.launchImageLibraryAsync({
      mediaTypes: ['images'],
      allowsEditing: true,
      quality: 1,
    });

    if (!result.canceled) {
      console.log(result);
    } else {
      alert('You did not select any image.');
    }
  };

  // ...rest of the code remains same
}
```

Пояснення:

- `launchImageLibraryAsync(options)` приймає об’єкт **ImagePickerOptions**.
- Якщо `allowsEditing: true`, користувач може кадрувати зображення під час вибору (iOS/Android).

3) Оновити компонент кнопки

Натискання на **primary-кнопку** має викликати `pickImageAsync()`. Додайте проп `onPress` до кнопки.

`components/Button.tsx` (фрагмент):

```tsx
import { StyleSheet, View, Pressable, Text } from 'react-native';
import FontAwesome from '@expo/vector-icons/FontAwesome';

type Props = {
  label: string;
  theme?: 'primary';
  onPress?: () => void;
};

export default function Button({ label, theme, onPress }: Props) {
  if (theme === 'primary') {
    return (
      <View
        style={[
          styles.buttonContainer,
          { borderWidth: 4, borderColor: '#ffd33d', borderRadius: 18 },
        ]}>
        <Pressable style={[styles.button, { backgroundColor: '#fff' }]} onPress={onPress}>
          <FontAwesome name="picture-o" size={18} color="#25292e" style={styles.buttonIcon} />
          <Text style={[styles.buttonLabel, { color: '#25292e' }]}>{label}</Text>
        </Pressable>
      </View>
    );
  }

  return (
    <View style={styles.buttonContainer}>
      <Pressable style={styles.button} onPress={() => alert('You pressed a button.')}>
        <Text style={styles.buttonLabel}>{label}</Text>
      </Pressable>
    </View>
  );
```

4) Використати обране зображення

У компоненті перегляду зображення перевіряємо: якщо користувач обрав картинку, показуємо її; інакше — плейсхолдер. (У прикладі застосунок **StickerSmash** використовує компонент з `expo-image`.)

`app/(tabs)/index.tsx` (фрагмент — підключаємо кнопку і передаємо хендлер):

```
<View style={styles.footerContainer}>
  <Button theme="primary" label="Choose a photo" onPress={pickImageAsync}
  <Button label="Use this photo" />
</View>
```

Використання вибраного фото

Тепер, коли користувач може обрати зображення з галереї, потрібно **зберегти його URI** (шлях до файлу) і **показати на екрані** замість плейсхолдера.

Крок 1. Імпорт і створення стану (useState)

React-компоненти мають стан — змінні, що зберігають дані між рендерами.
Ми використаємо хук `useState`, щоб зберегти шлях до вибраного фото.

У файлі **`app/(tabs)/index.tsx`**:

```
import { View, StyleSheet } from 'react-native';
import * as ImagePicker from 'expo-image-picker';
import { useState } from 'react';

import Button from '@/components/Button';
import ImageViewer from '@/components/ImageViewer';

const PlaceholderImage = require('@/assets/images/background-image.png');

export default function Index() {
  const [selectedImage, setSelectedImage] = useState<string | undefined>(undefined);

  const pickImageAsync = async () => {
    let result = await ImagePicker.launchImageLibraryAsync({
      mediaTypes: ['images'],
      allowsEditing: true,
      quality: 1,
    });

    if (!result.canceled) {
      setSelectedImage(result.assets[0].uri);
    } else {
      alert('You did not select any image.');
    }
  };

  return (
    <View style={styles.container}>
      <View style={styles.imageContainer}>
        <ImageViewer imgSource={PlaceholderImage} selectedImage={selectedImage} />
      </View>
      <View style={styles.footerContainer}>
        <Button theme="primary" label="Choose a photo" onPress={pickImageAsync} />
        <Button label="Use this photo" />
      </View>
    </View>
  );
}
```

Крок 2. Оновлення компонента `ImageViewer`

Тепер компонент `ImageViewer` має приймати два пропси:

- `imgSource` — стандартне зображення (плейсхолдер із `assets`);
- `selectedImage` — URI вибраного користувачем фото.

Файл `components/ImageViewer.tsx`:

```tsx
import { ImageSourcePropType, StyleSheet } from 'react-native';
import { Image } from 'expo-image';

type Props = {
  imgSource: ImageSourcePropType;
  selectedImage?: string;
};

export default function ImageViewer({ imgSource, selectedImage }: Props) {
  const imageSource = selectedImage ? { uri: selectedImage } : imgSource;

  return <Image source={imageSource} style={styles.image} />;
}

const styles = StyleSheet.create({
  image: {
    width: 320,
    height: 440,
    borderRadius: 18,
  },
});
```

Крок 3. Як це працює

1. Коли користувач натискає кнопку **“Choose a photo”**, викликається `pickImageAsync()`.
2. Відкривається системний пікер (галерея телефону).
3. Якщо користувач вибрав фото — ми зберігаємо його шлях у стані:

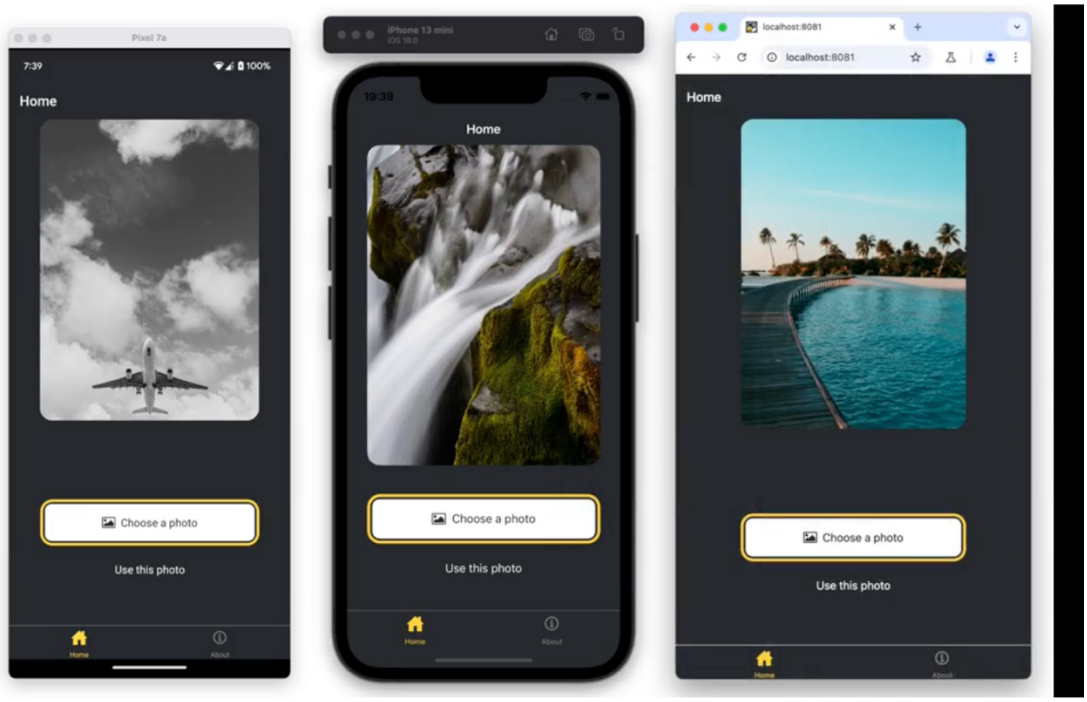
   ```
   setSelectedImage(result.assets[0].uri);
   ```

4. React перерендерює компонент → `selectedImage` тепер має URI фото.
5. `ImageViewer` отримує новий проп `selectedImage` і показує фото замість плейсхолдера.

Крок 4. Результат

Тепер застосунок:

- відображає плейсхолдер за замовчуванням;
- відкриває галерею після натискання на “Choose a photo”;
- показує вибране фото на екрані.


Pixel 7a
7:39
100%
Home
Choose a photo
Use this photo
Home
About
iPhone 13 mini
iOS 18.0
19:39
Home
Choose a photo
Use this photo
Home
About
localhost:8081
localhost:8081
Home
Choose a photo
Use this photo
Home
About