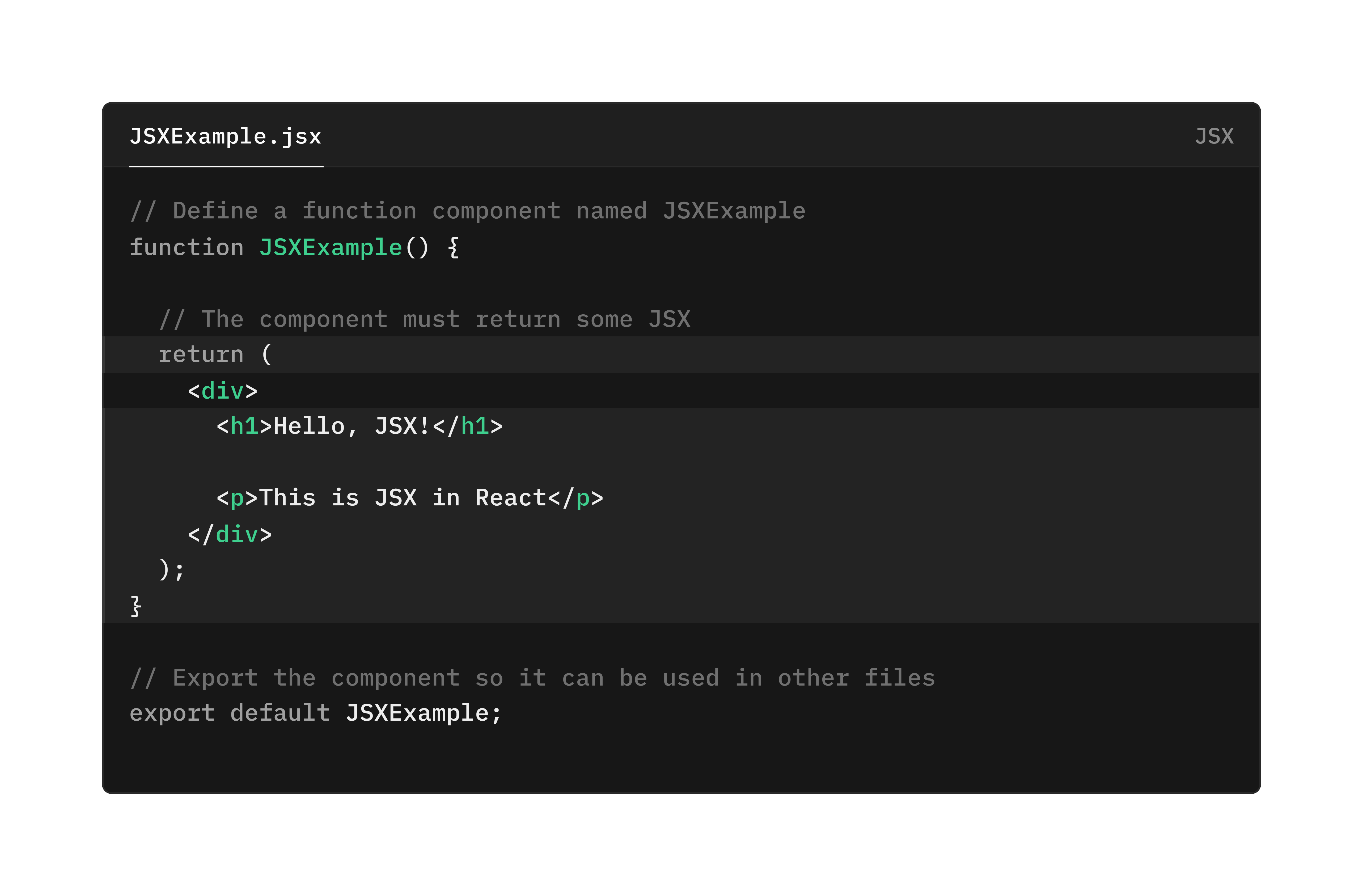
**Basic Understanding In React**

**React பற்றிய அடிப்படை புரிதல்**

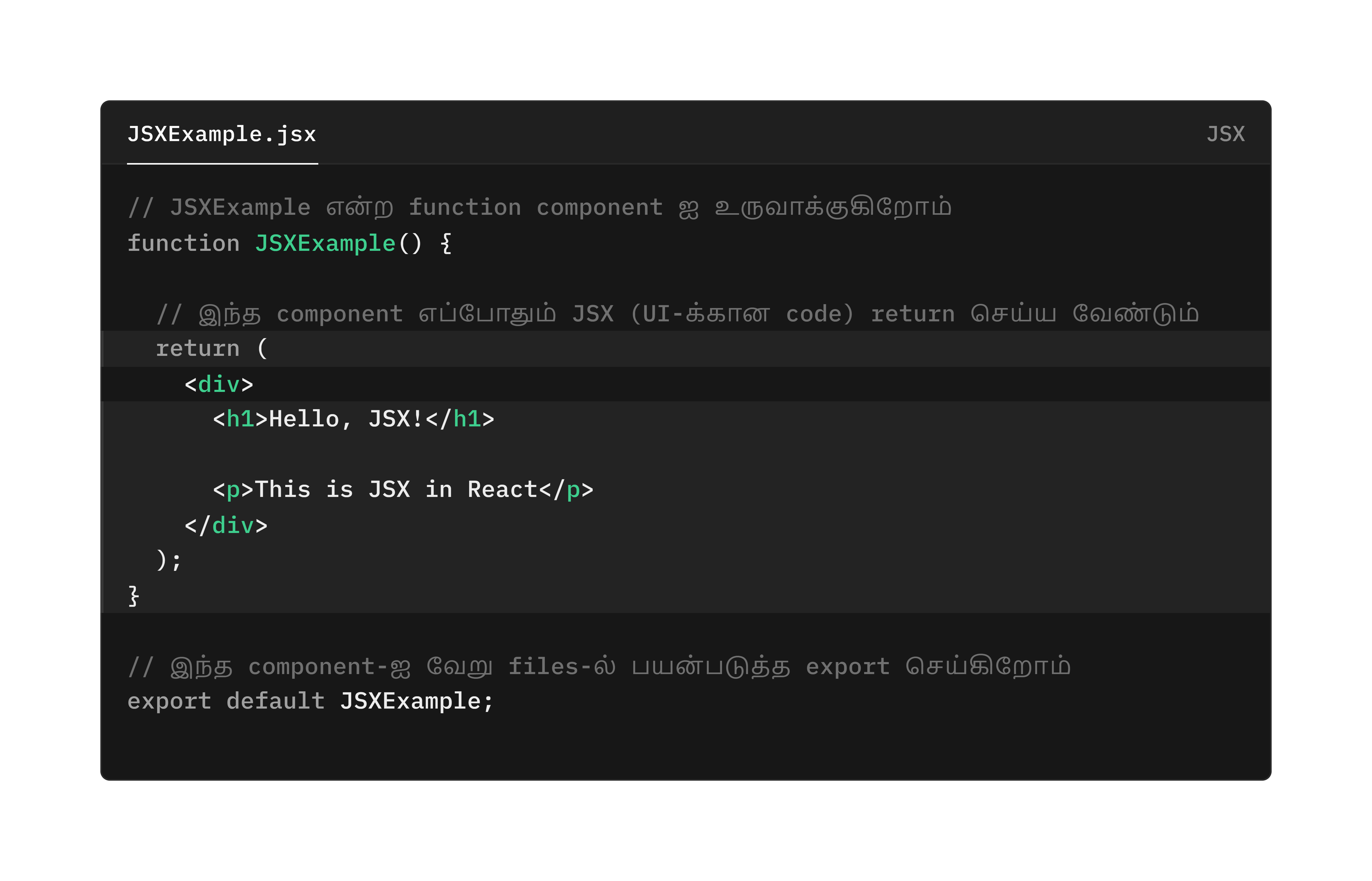
**1.JSX**

* JSX allows us to write **HTML-like code inside JavaScript**.
* JSX, JavaScript-க்குள் HTML போன்ற code எழுத அனுமதிக்கிறது.
* It makes UI design **easy and readable**.
* இது UI வடிவமைப்பை எளிதாகவும் படிக்கக்கூடியதாகவும் ஆக்குகிறது.
* JSX must always have **one parent element** wrapping everything.
* JSX எப்போதும் எல்லாவற்றையும் உள்ளடக்கிய ஒரு பெற்றோர் கூறுகளைக் கொண்டிருக்க வேண்டும்.

**Example:**



**உதாரணமாக:**



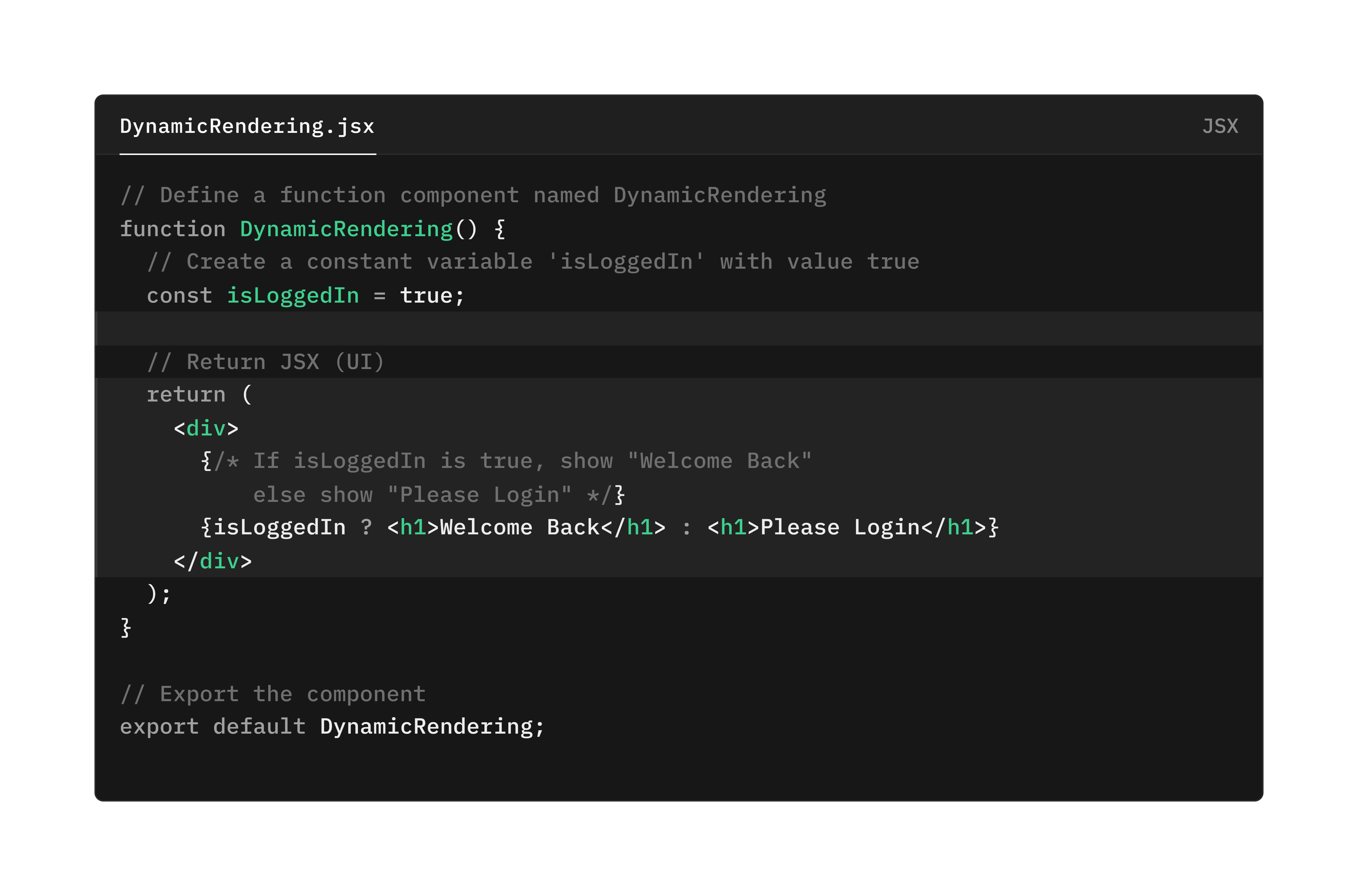
**Explanation:**

* The return has one parent <div> wrapping <h1> and <p>.
* ரிட்டர்னில் <div> <h1> மற்றும் <p> என ஒரு பேரன்ட் உள்ளது.
* Without the parent, JSX will throw an error.
* பேரன்ட் இல்லாமல், JSX ஒரு பிழையை ஏற்படுத்தும்.
* JSX looks like HTML but actually becomes **JavaScript objects** after compilation.
* JSX HTML போல தோற்றமளிக்கும் ஆனால் தொகுத்தலுக்குப் பிறகு உண்மையில் ஜாவாஸ்கிரிப்ட் பொருள்களாக மாறும்.

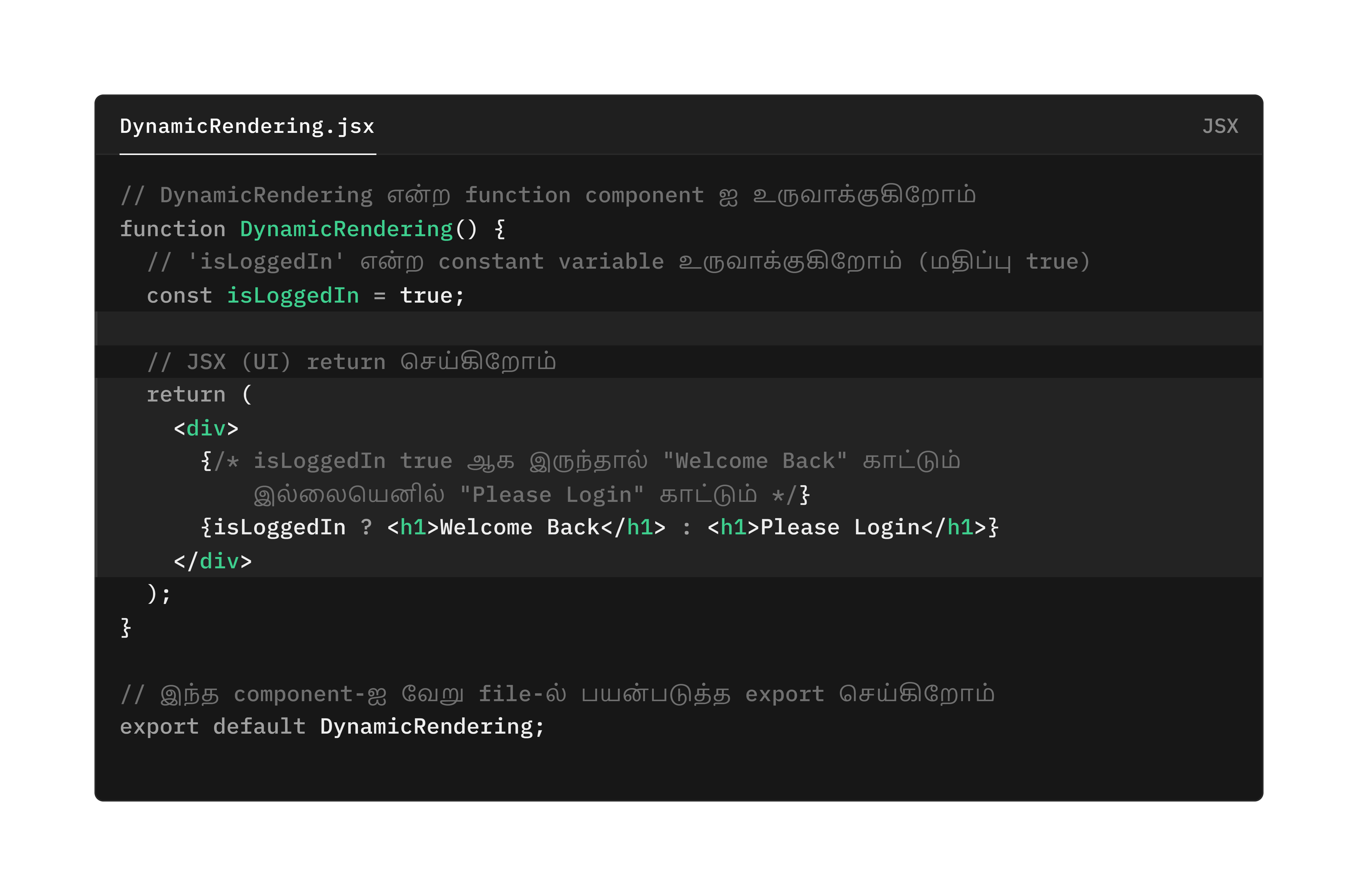
**2. Dynamic Rendering in React**

* Dynamic rendering means the **UI updates automatically** when data changes.
* டைனமிக் ரெண்டரிங் என்பது தரவு மாறும்போது UI தானாகவே புதுப்பிக்கப்படுவதைக் குறிக்கிறது.
* React updates only the **changed parts**, making it very fast.
* ரியாக்ட் மாற்றப்பட்ட பகுதிகளை மட்டுமே புதுப்பிக்கிறது, இது மிக வேகமாகிறது.
* **Common methods** ( பொதுவான முறைகள் ):
* Using **ternary operator** (condition ? value1 : value2)
* Using **if statements**

**Example:**



**உதாரணமாக:**



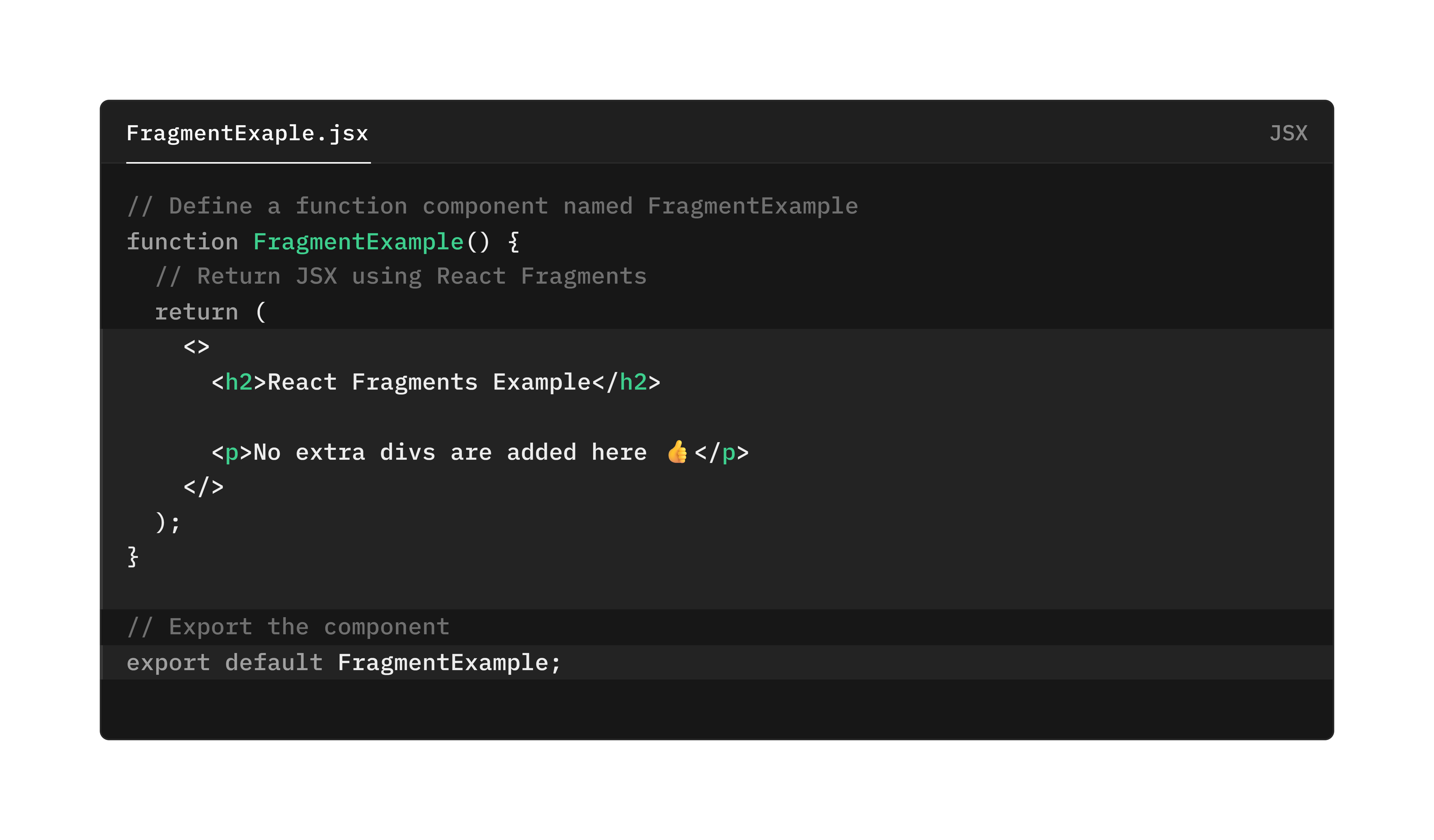
**Explanation:**

* {isLoggedIn ? ... : ...} → Ternary operator used inside JSX.
* {isLoggedIn ? ... : ...} → JSX-க்குள் பயன்படுத்தப்படும் டெர்னரி ஆபரேட்டர்.
* If isLoggedIn is true → shows **Welcome Back**.
* isLoggedIn உண்மையாக இருந்தால் → மீண்டும் வரவேற்கிறேன் என்பதைக் காட்டுகிறது.
* If false → shows **Please Login**.
* தவறு என்றால் → தயவுசெய்து உள்நுழையவும் என்பதைக் காட்டுகிறது.
* React updates only the changed text, not the whole page.
* ரியாக்ட் முழு பக்கத்தையும் அல்ல, மாற்றப்பட்ட உரையை மட்டுமே புதுப்பிக்கிறது.

**3. Fragments in React**

* In React, every component must return **a single parent element**.
* React-இல், ஒவ்வொரு கூறும் ஒரு ஒற்றை பெற்றோர் உறுப்பைத் திருப்பி அனுப்ப வேண்டும்.
* If we don’t want to add extra <div> wrappers, we use **Fragments**.
* கூடுதல் <div> ரேப்பர்களைச் சேர்க்க விரும்பவில்லை என்றால், Fragments-ஐப் பயன்படுத்துகிறோம்.
* Fragments prevent unnecessary DOM elements, making HTML cleaner and performance better.
* Fragments தேவையற்ற DOM கூறுகளைத் தடுக்கிறது, இதனால் HTML சுத்தமாகவும் செயல்திறனை மேம்படுத்தவும் உதவுகிறது.

**Example:**



**உதாரணமாக :**



**Explanation:**

* <> ... </> → Short syntax for **React.Fragment**.
* <> … </> → React.Fragment க்கான குறுகிய தொடரியல்.
* Groups <h2> and <p> together without adding a <div>.
* <div> ஐ சேர்க்காமல் குழுக்கள் <h2> மற்றும் <p> ஐ ஒன்றாக இணைக்கவும்.
* Final HTML won’t have extra wrappers.
* இறுதி HTML இல் கூடுதல் ரேப்பர்கள் இருக்காது.

**4. Components in React**

* A **component** is a **reusable block** in React.
* ஒரு கூறு என்பது React இல் மீண்டும் பயன்படுத்தக்கூடிய தொகுதி.
* Components work like **functions that return JSX**.
* கூறுகள் JSX ஐத் திருப்பி அனுப்பும் செயல்பாடுகளைப் போலவே செயல்படுகின்றன.
* They make UI easy to **build and manage**.
* அவை UI ஐ உருவாக்கவும் நிர்வகிக்கவும் எளிதாக்குகின்றன.

**Types of Components:**

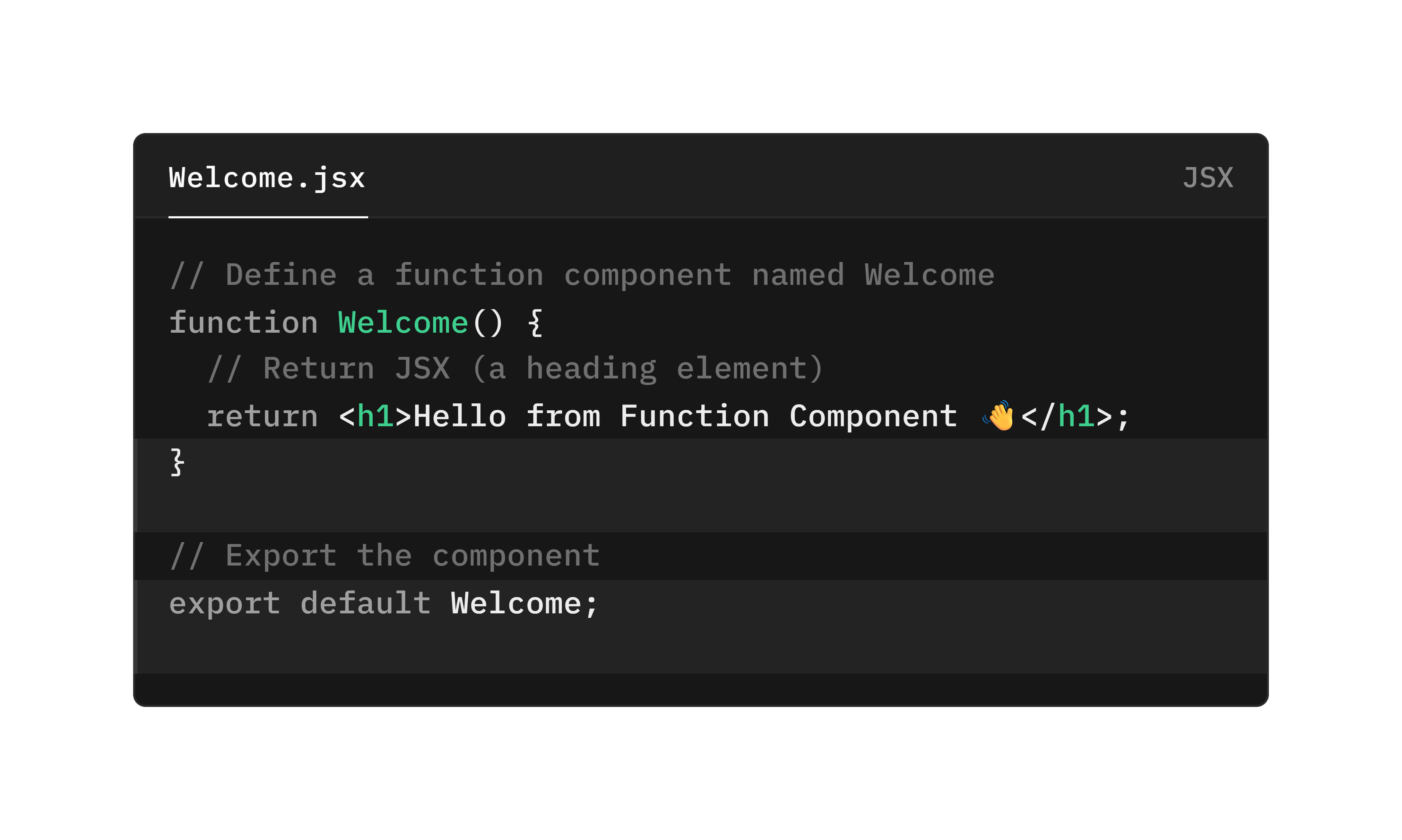
1. **Function Components –** simple and commonly used.

**(**எளிமையானவை மற்றும் பொதுவாகப் பயன்படுத்தப்படுகின்றன.**)**

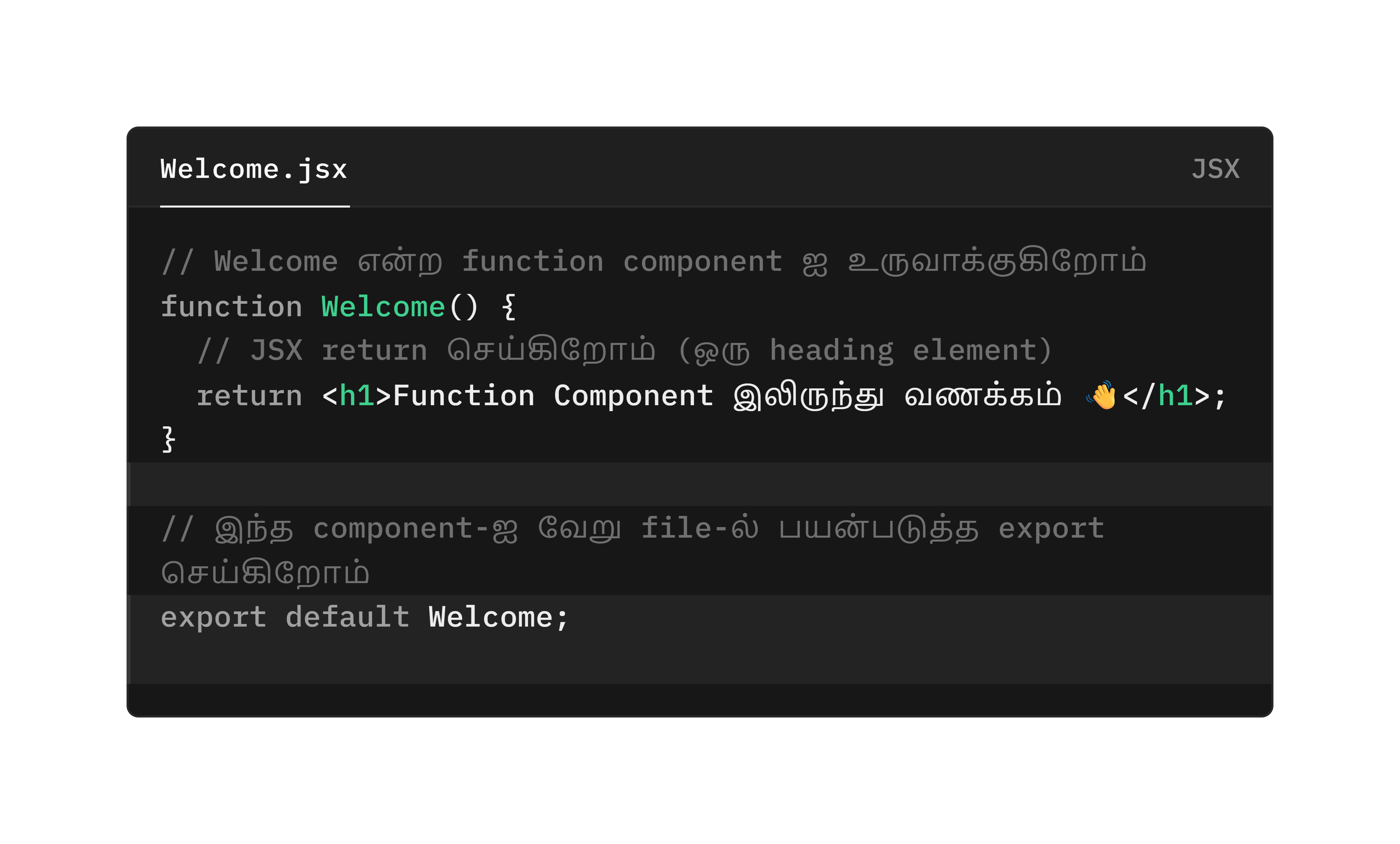
1. **Class Components** – older style, less used now but still important.

**(**பழைய பாணி, இப்போது குறைவாகப் பயன்படுத்தப்படுகிறது, ஆனால் இன்னும் முக்கியமானது. **)**

**Example – Function Component:**



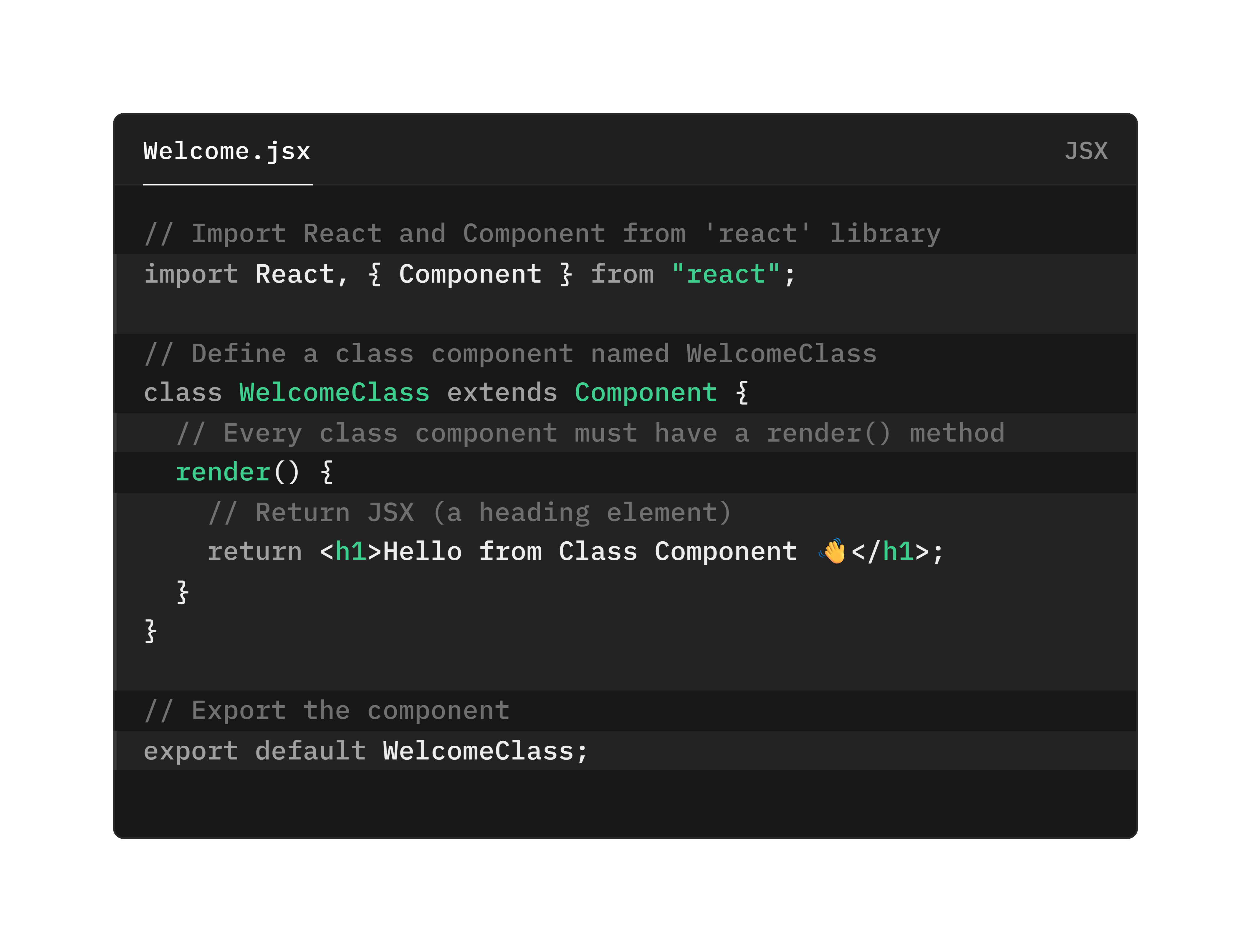
**உதாரணமாக:**



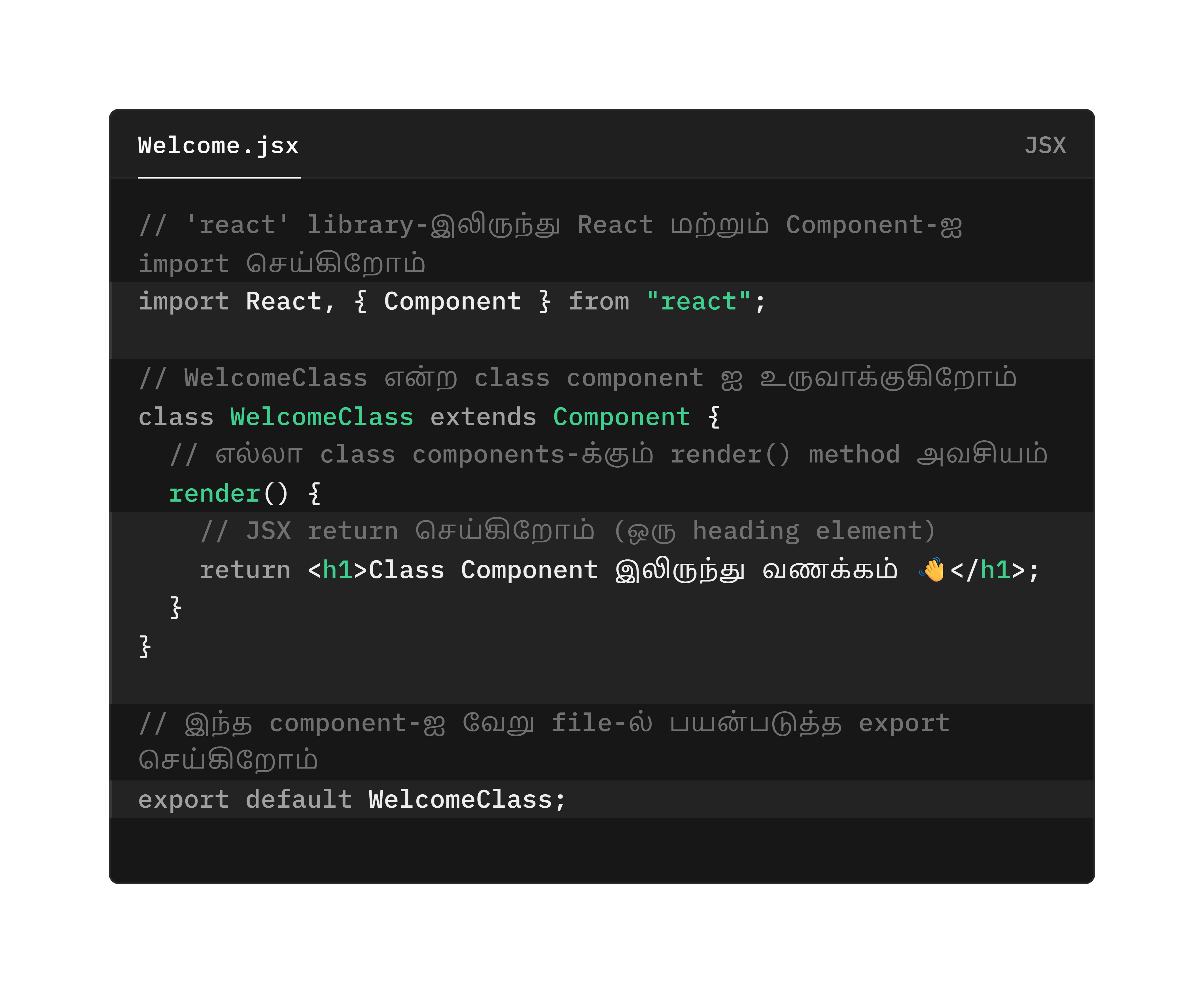
Easy to write, uses plain functions.

எழுத எளிதானது, எளிய செயல்பாடுகளைப் பயன்படுத்துகிறது.

**Example – Class Component:**



**உதாரணமாக:**



Uses ES6 classes, render() method returns JSX.

ES6 வகுப்புகளைப் பயன்படுத்துகிறது, ரெண்டர்() முறை JSX ஐ வழங்குகிறது.