# JSX

## Overview

In this lab you'll simplify the "library" web page from the previous lab, so that it uses JSX syntax rather than calling React.createElement() everywhere.

You can either choose to implement your components as class-based components or as functional components. We provide starter code and solutions for both techniques.

## Source folders

* ReactDev\Student\04-JSX
* ReactDev\Solutions\04-JSX

## Roadmap

Here's a brief summary of the exercises in this lab. More detailed instructions follow later in this lab document:

1. Familiarization with the 'solution' web pages
2. Getting started with the 'student' web page
3. Preparing to use JSX
4. Refactoring the code to use JSX

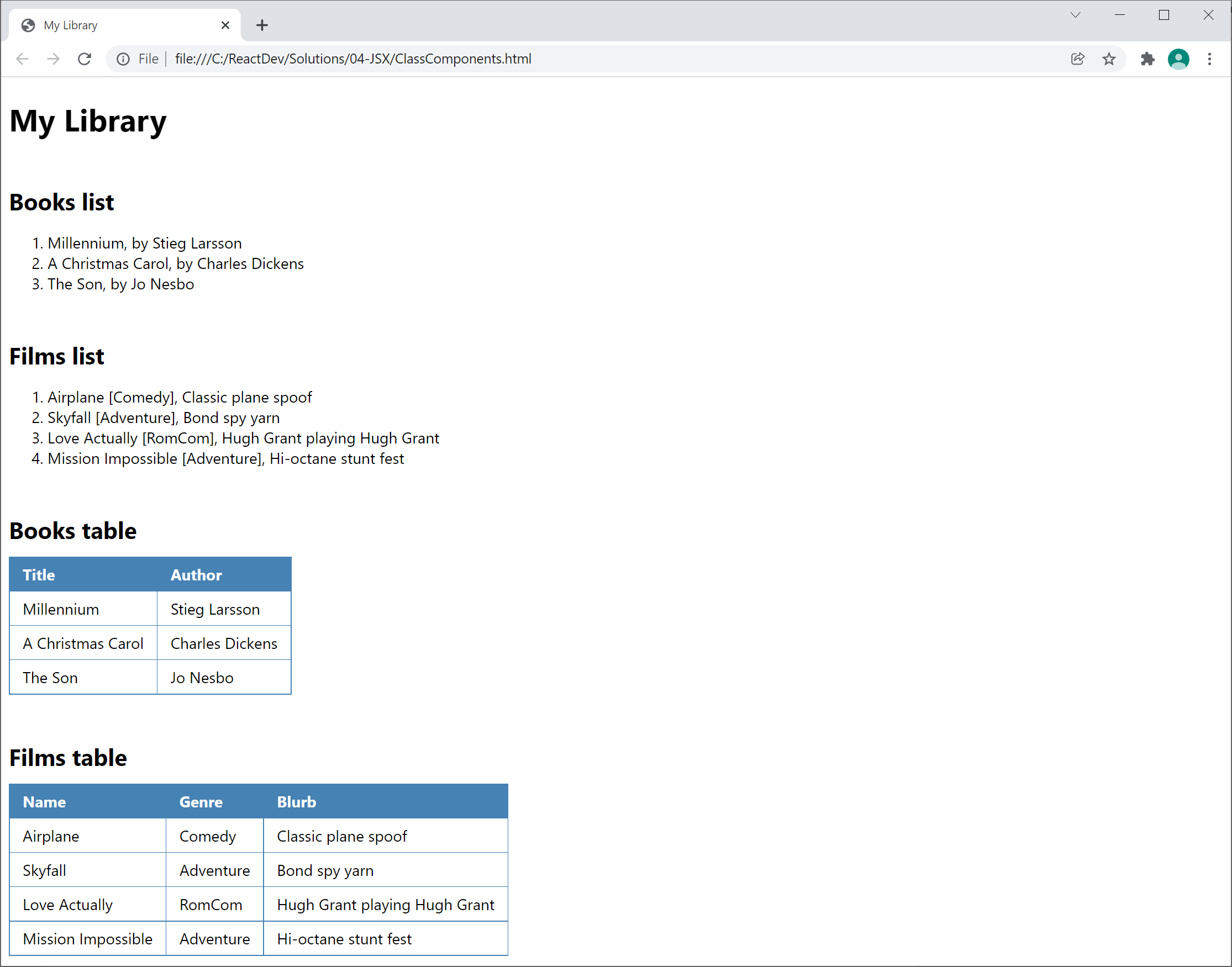
## Exercise 1: Familiarization with the 'solution' web pages

Open either of the following web pages in a browser:

ReactDev\Solutions\04-JSX\ClassComponents.html

ReactDev\Solutions\04-JSX\FunctionalComponents.html

These two web pages are semantically equivalent, and the UI is the same as in the previous lab:



## Exercise 2: Getting started with the 'student' web page

Now go to the ***student*** folder and open either of the following web pages in a text editor:

ReactDev\Student\04-JSX\ClassComponents.html

ReactDev\Student\04-JSX\FunctionalComponents.html

These files are the same as the solutions from the previous lab, including the "if time permits" bits. Take a moment to get familiar with the code.

## Exercise 3: Preparing to use JSX

The way we're using JSX at the moment, you have to explicitly add support for Babel in the web page. This enables the browser to use Babel to transpile JSX code into pure Java when the page is loaded.

With this in mind, make the following 2 changes:

* Add a <script> tag to import the Babel library.
* In the main <script> tag for your web page, add a type="text/babel" property so Babel knows it has to transpile the code contained in this script section.

## Exercise 4: Refactoring the code to use JSX

Refactor the code in the web page to use JSX rather than calling React.createElement() everywhere. We suggest you refactor one component at a time, and test everything still works as you go along.

Here are a few things to bear in mind:

* JSX uses XML syntax, so tag names are case-sensitive and every start tag must have a corresponding end tag.
* If you want to evaluate a JavaScript expression within an XML tag, you must enclose the JavaScript expression inside {} braces.