Declarative

Arnav Gupta

April 19, 2024

Contents

1 UI Programming Paradigms

Imperative is describing *how* to achieve a result. Declarative is describing *what* the desired result is.

SimpleKit used imperative to build a UI (tree of nodes with events, code describes how to make that tree).

HTML can be declarative (tree of nodes with events, but markup describes the desired tree/DOM).

Some DOM manipulation methods mix declarative and imperative UI building with HTML for declarative (layout) and TypeScript for imperative (events or modifying layout).

2 Declarative Syntax

HyperScript is a language to generate descriptions of UI trees. `hyperscript` is an npm package to generate HyperScript.

HyperScript function calls create a representation of the UI tree, which is a JS object. This is commonly referred to as a virtual DOM.

HTM is syntactic sugar for HyperScript.

3 Components and Virtual DOM

Virtual DOM is used for 2 purposes:

1. render an actual DOM using imperative methods
2. lightweight abstraction of DOM to compare changes, which enables efficient DOM diffing for reactivity

4 Preact and JSX

Library: a collection of code (types, objects, functions, etc)

- can use parts in a flexible way
- call library code
- unopinionated collections of code
- ex. React, Preact, Bootstrap

Framework: a structure to use a collection of provided code

- modify and extend, but follow a set of rules
- framework calls code (inversion of control)
- frameworks are opinionated libraries
- ex. Angular, NextJS

Toolkit: a collection of compatible libraries, such as Bootstrap

Tool: software to perform a certain task, such as Vite

Preact is not a reimplementation of React. Differences are:

- Preact uses native DOM elements, where React has its own synthetic event system
  - quirks: onInput vs onChange, onDblClick vs onDoubleClick
- Preact treats children nodes as native JS arrays, where React has its own object for managing children
- Preact supports `class` to set class attribute, where React uses `className`

4.1 JSX

Describe DOM trees with a mixture of JS and HTML. Files with JSX are compiled into JS (hyperscript function calls).

How JSX works:

1. JSX is compiled into hyperscript
2. hyperscript is used to render
   (a) `h` function to create a virtual DOM object
   (b) `render` function to create DOM from virtual DOM

4.2 Components

Components are the building blocks of a Preact application, and have custom properties (props).

Functions are the most common way to create components.

Components can also be defined as classes, though this is no longer common and functional components are better.

Components can be nested like HTML elements, and can have HTML or components nodes as children. Enables control over how virtual DOM elements nested witin a component should be rendered. Array of children is a special implicit prop.

For TypeScript, best practice is to define a props type for a component, which can have optional props.

Props can be destructured in the functional component header to make it easier to assign default prop values.

Preact uses standard DOM events with declarative syntax. If the even handler is small, the function can be defined inline. Else, a handler function should be called.

Event handlers can be passed as props to components.

An **expression** is a valid unit of code that resolves to a value.

JSX is an expression, and everything in JSX must be an expression, using curly braces. For arrays, use map. For conditional logic, use the ternary operator.

For dynamic attributes, either use template literal or hyperscript object.

If code does not naturally have a single root node, wrap component nodes in `<Fragment>` or `<>` tags, which are not rendered in the DOM.