By Maxime Deschamps - Submitted on 24/01/2022

Order Book Assessment - Additional Notes

1. OrderBook Component

Under the components section, the orderBook folder contains all elements needed for the component. The root <OrderBook / > component, the subcomponent folder, and models and hooks folders containing elements that are specific to the OrderBook. Next to the OrderBook component, an OrderBookHelpers file containing pure functions used by the component. These are where the heavier calculations are done.
The Hooks folder inside the orderBook folder contains the useOrderBookFeed hook, as well as the useOrderBookFeedHelper and OrderBookWebSocket class, both used by the hook.
All components, helpers and the hook have tests in adjacent folders.

2. The useOrderBookFeed hook and OrderBookWebSocket class

The useOrderBookFeed hook is responsible for creating the OrderBookWebSocket instance when the product_id (‘pair’) is toggled. The hook also tells the OrderBookWebSocket instance how to handle and parse a websocket message to its asks and bids state to serve to the root component later on. The hook is also responsible for telling the OrderBookWebSocket when to disconnect and reconnect.
The OrderBookWebSocket class creates a new WebSocket using constructor parameters, and exposes init() and close() functions.

This separation is good for expandability. The OrderBookWebSocket class could be used by a useOrderHistoryFeed hook that stores the orders in a different manner and requires messages to be handled differently.

3. Render throttling

Inside the OrderBook component, the throttledOrderBookRows const contains the orderbook data for the asks, bids, and spread info, and does the heavy lifting to calculate the Components to render:

```
const throttledOrderBookRows = useMemo(
    () => {
      if (!asks.length && !bids.length)
        return <Row>
          <Col className={styles.noConnection}>No websocket
connection</Col>
        </Row>;
      ...
    },
    [throttleTick]
  );
```

This useMemo hook uses the throttleTick to decide when to rerender a new version, This way, this section does not rerender every time something changes to the asks or bids collections on the state.
The throttleTick state comes from the useThrottleTick hook, which uses the useInterval hook to decide when to cause an invalidation of the memoed throttleOrderBookRows.

4. <OrderBookSettings />

This component exposes the depth and throttleInterval states with slider inputs. This is a development tool I added to help the optimisation process. It is very useful when optimizing for optimal FPS, and best CPU and memory use. It also illustrated the flexibility of the OrderBook component in real time.
Default depth and throttleInterval values can be passed as props to the OrderBook, as well as a showDevTools prop that enables the OrderBookSettings component externally.
In this example, a defaultDepth of 15 and defaultThrottleInterval of 350 ms gives a good balance between performance and user experience, not too fast for the load on the system, and not too fast for the user experience.

5. Orderbook Focus Handling

When the user leaves the browser window, the window.onblur handler inside the useOrderBookFeed hook unsubscribes and disconnects from the WebSocket. A modal is then shown, showing the option to reconnect.

```
const OrderBookPage = () => {
  return (
    <ErrorBoundary FallbackComponent={ErrorFallback}
onError={myErrorHandler}>
      <OrderBook
        defaultDepth={15}
        defaultThrottleInterval={350}
        showDevTools
      />
    </ErrorBoundary>
  )
}
```

6. Test with React Testing Library and Jest

Functional and UI tests were written using RTL and Jest to test all major components and functional logic. All orderBook components also have RTL Snapshot tests on top of the UI tests to ensure the best stability.

7. Project Structure

```
│   .eslintrc.json
│   .swcrc
│   Constants.js
│   jest.config.js
│   jest.setup.ts
│   next-env.d.ts
│   next.config.js
│   package.json
│   README.md
│   tree.txt
│   tsconfig.json
│
├───components
│   ├───common
│   │   │   Layout.tsx
│   │   │
│   │   └───hooks
│   │           index.js
│   │           useInterval.tsx
│   │           useThrottleTick.ts
│   │
│   └───orderBook
│       │   OrderBook.module.scss
│       │   OrderBook.tsx
│       │   OrderBookHelpers.ts
│       │
│       ├───hooks
│       │   │   index.js
│       │   │   OrderBookWebSocket.tsx
│       │   │   useOrderBookFeed.ts
│       │   │   useOrderBookFeedHelpers.tsx
│   │   │   WebSocketOutMessage.ts
│   │   │
│   │   └───tests
│   │           useOrderBookFeed.test.ts
│   │           useOrderBookFeedHelper.test.ts
│   │
│   ├───models
│   │       Order.ts
│   │       OrderType.ts
│   │       Spread.ts
│   │
│   ├───subcomponents
│   │   │   FeedToggle.tsx
│   │   │   OrderBookHeader.tsx
│   │   │   OrderBookSettings.tsx
│   │   │   OrderRow.tsx
│   │   │   OrderRowNumbers.tsx
│   │   │   ReconnectModal.tsx
│   │   │
│   │   └───tests
│   │       │   FeedToggle.test.js
│   │       │   OrderBookHeader.test.js
│   │       │   OrderBookSettings.test.js
│   │       │   OrderRow.test.js
│   │       │   OrderRowNumbers.test.js
│   │       │   ReconnectModal.test.js
│   │       │
│   │       └───__snapshots__
│   │               FeedToggle.test.js.snap
│   │               OrderBookHeader.test.js.snap
│   │               OrderBookSettings.test.js.snap
│   │               OrderRow.test.js.snap
│   │               OrderRowNumbers.test.js.snap
│   │
│   └───tests
│       │   OrderBook.test.js
│       │   OrderBookHelpers.test.js
│       │
│       └───__snapshots__
│               OrderBook.test.js.snap
│
├───pages
│       index.tsx
│       _app.tsx
│
├───public
│       favicon.ico
│       vercel.svg
│
├───styles
│       globals.scss
│
└───tools
        orderBy.ts
```