



Redux Cheatsheet(Part 1)

- **Store:** Centralized state container.
 - Example: `const store = createStore(reducer);`
- **Reducer:** Function to handle state changes.
 - Example: `function counterReducer(state, action) { ... }`
- **Action:** Object describing a state change.
 - Example: `{ type: 'INCREMENT' }`
- **Dispatch:** Send actions to the store.
 - Example: `store.dispatch({ type: 'INCREMENT' });`
- **getState:** Retrieve current state. www.developerupdates.com
 - Example: `const currentState = store.getState();`
- **subscribe:** Listen for state changes.
 - Example: `store.subscribe(() => { ... });`
- **combineReducers:** Combine multiple reducers.
 - Example: `combineReducers({ user: userReducer, posts: postsReducer });`
- **applyMiddleware:** Add middleware to the store.
 - Example: `applyMiddleware(logger, thunk);`
- **Middleware:** Intercept actions for processing.
 - Example: `const logger = store => next => action => { ... };`
- **Thunk:** Middleware for async actions.
 - Example: `const fetchPosts = () => dispatch => { ... };`



Chetan Mahajan

www.developerupdates.com



Are you looking for a Front-end Developer Job? If yes, Check the link in bio to get Interview Kit.



Redux Cheatsheet(Part 2)

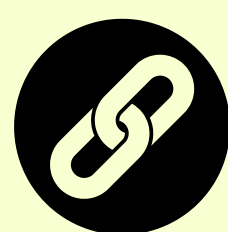
- **createAction**: Generate action creators.
 - Example: **const increment = createAction('INCREMENT');**
- **createReducer**: Simplify reducer creation.
 - Example: **const counterReducer = createReducer(initialState, { ... });**
- **Provider**: Pass store to React components.
 - Example: **<Provider store={store}>...</Provider>**
- **connect**: Connect React components to Redux.
 - Example: **connect(mapStateToProps, mapDispatchToProps)(Component);**
- **mapStateToProps**: Map state to component props.
 - Example: **const mapStateToProps = state => ({ count: state.count });**
- **mapDispatchToProps**: Map dispatch to component props.
 - Example: **const mapDispatchToProps = dispatch => ({ ... });**
- **useSelector**: Access state in functional components.
 - Example: **const count = useSelector(state => state.count);**
- **useDispatch**: Dispatch actions in functional components.
 - Example: **const dispatch = useDispatch();**
- **useStore**: Access the Redux store directly.
 - Example: **const store = useStore();**
- **combineReducers**: Combine multiple reducers.
 - Example: **combineReducers({ user: userReducer, posts: postsReducer });**

www.developerupdates.com



Chetan Mahajan

www.developerupdates.com



Are you looking for a Front-end Developer Job? If yes, Check the link in bio to get Interview Kit.



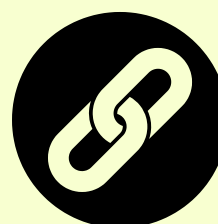
Redux Cheatsheet(Part 3)

- **configureStore**: Simplify store configuration.
 - Example: **const store = configureStore({ reducer: rootReducer });**
- **createSlice**: Create a Redux slice with actions and reducers.
 - Example: **const slice = createSlice({ name, initialState, reducers });**
- **initialState**: Default state value.
 - Example: **const initialState = { count: 0 };**
- **payload**: Data carried by an action.
 - Example: **{ type: 'ADD_TODO', payload: { text: 'Learn Redux' } }**
- **DevTools**: Redux browser extension for debugging.
 - Example:
composeWithDevTools(applyMiddleware(...middlewares));
www.developerupdates.com
- **Immer**: Simplify immutable updates.
 - Example: **import produce from 'immer';**
- **Selector**: Extract data from state.
 - Example: **const selectTodos = state => state.todos;**
- **Normalized State**: Flattened state shape.
 - Example: **{ entities: { users: {}, posts: {} }, ids: [] }**
- **createAsyncThunk**: Handle async actions in Redux Toolkit.
 - Example: **const fetchUser = createAsyncThunk('users/fetch', async id => { ... });**



Chetan Mahajan

www.developerupdates.com



Are you looking for a Front-end Developer Job? If yes, Check the link in bio to get Interview Kit.



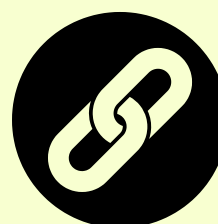
Redux Cheatsheet(Part 4)

- **batch**: Dispatch multiple actions together.
 - Example: `batch(() => { dispatch(action1); dispatch(action2); });`
- **action creators**: Functions that return action objects.
 - Example: `const addTodo = text => ({ type: 'ADD_TODO', payload: text });`
- **Action Types**: Constants for action type strings.
 - Example: `const INCREMENT = 'INCREMENT';`
- **Redux Observable**: RxJS-based middleware for handling async actions.
 - Example: `const epic = action$ => action$.pipe(ofType('PING'), mapTo({ type: 'PONG' }));` www.developerupdates.com
- **Thunk Middleware**: Middleware for handling functions as actions.
 - Example: `const middleware = [thunk];`
- **Store Subscription**: Execute a function when state changes.
 - Example: `store.subscribe(() => { console.log(store.getState()); });`
- **Lazy Initialization**: Initialize state only when needed.
 - Example: `const [state, dispatch] = useReducer(reducer, initialState, init);`
- **Redux Logger**: Middleware for logging actions and state.
 - Example: `const logger = createLogger();`
- **Thunk Dispatch**: Dispatch function returned by thunk action creator.
 - Example: `dispatch(fetchUser());`



Chetan Mahajan

www.developerupdates.com



Are you looking for a Front-end Developer Job? If yes, Check the link in bio to get Interview Kit.