Awesome Redux cheatsheet

redux-actions

Create action creators in flux standard action format.

```
increment = createAction('INCREMENT', amount => amount)
increment = createAction('INCREMENT')  // same
```

```
increment(42) === { type: 'INCREMENT', payload: 42 }
```

```
// Errors are handled for you:
err = new Error()
increment(err) === { type: 'INCREMENT', payload: err, error: true }
```

redux-actions

reduce-reducers

Combines reducers (like combineReducers()), but without namespacing magic.

```
re = reduceReducers(
  (state, action) => state + action.number,
  (state, action) => state + action.number
)

re(10, { number: 2 })  //=> 14
```

reduce-reducers

flux-standard-action

A standard for flux action objects. An action may have an `error`, `payload` and `meta` and nothing else.

```
{ type: 'ADD_TODO', payload: { text: 'Work it' } }
{ type: 'ADD_TODO', payload: new Error(), error: true }
```

flux-standard-action

redux-multi

Dispatch multiple actions in one action creator.

```
store.dispatch([
  { type: 'INCREMENT', payload: 2 },
  { type: 'INCREMENT', payload: 3 }
])
```

redux-multi

redux-logger

Logs actions to your console.

```
// Nothing to see here
```

redux-logger

Async

redux-promise

Pass promises to actions. Dispatches a flux-standard-action.

```
increment = createAction('INCREMENT')  // redux-actions
increment(Promise.resolve(42))
```

redux-promise

redux-effects

Pass side effects declaratively to keep your actions pure.

```
{
  type: 'EFFECT_COMPOSE',
  payload: {
    type: 'FETCH'
    payload: {url: '/some/thing', method: 'GET'}
  },
  meta: {
    steps: [ [success, failure] ]
  }
}
```

redux-effects

redux-promises

Sorta like that, too. Works by letting you pass thunks (functions) to `dispatch()`. Also has 'idle checking'.

```
fetchData = (url) => (dispatch) => {
  dispatch({ type: 'FETCH_REQUEST' })
  fetch(url)
    .then((data) => dispatch({ type: 'FETCH_DONE', data })
    .catch((error) => dispatch({ type: 'FETCH_ERROR', error })
})

store.dispatch(fetchData('/posts'))
```

```
// That's actually shorthand for:
fetchData('/posts')(store.dispatch)
```

redux-promises

redux-thunk

Pass "thunks" to as actions. Extremely similar to redux-promises, but has support for getState.

```
fetchData = (url) => (dispatch, getState) => {
  dispatch({ type: 'FETCH_REQUEST' })
  fetch(url)
    .then((data) => dispatch({ type: 'FETCH_DONE', data })
    .catch((error) => dispatch({ type: 'FETCH_ERROR', error })
})

store.dispatch(fetchData('/posts'))
```

```
// That's actually shorthand for:
fetchData('/posts')(store.dispatch, store.getState)
```

```
// Optional: since fetchData returns a promise, it can be chained
// for server-side rendering
store.dispatch(fetchPosts()).then(() => {
  ReactDOMServer.renderToString(<MyApp store={store} />)
})
```

redux-thunk

0 Comments for this cheatsheet. Write yours!