React ❤ TS

Advanced TypeScript Patterns

Marc Missey

Senior Engineer @Twilio Sendgrid

@marcmissey - Twitter
@marc - Denver Devs

React

TypeScript

Maintainability

Benefits to Maintainability

- Confidence in refactoring

Benefits to Maintainability

- Confidence in refactoring
- Intellisense in IDEs

Benefits to Maintainability

- Confidence in refactoring
- Intellisense in IDEs
- Prevents *"clever"* code mistakes

Benefits to Maintainability

- Confidence in refactoring
- Intellisense in IDEs
- Prevents *"clever"* code mistakes
- Easier to read and understand

But this talk is **not** a sales pitch

TypeScript **Nirvana** 🧘🏻‍♂️

TypeScript **Nirvana** 🧘🏻‍♂️

aka: Maximizing Type Inference

Demo App

github.com/mmissey/ts-base

Patterns For Strongly Typed...

- State Management
- Redux Connected Components
- Higher Order Components
- Miscellaneous

Strongly Typed *State Management*

ActionTypes -> Actions -> Side Effects -> Actions -> Reducer -> Containers -> Components

Code Example

Patterns For Strongly Typed...

- ✅ State Management
- Redux Connected Components
- Higher Order Components
- Miscellaneous

Redux Connected Components

```
import { connect } from 'redux';
import { login, logout } from './actionCreators';

const mapStateToProps = (state) => {
  return {
    user: state.user
  };
}
const mapDispatchToProps = (dispatch) => {
  return {
    login: (userId: string) => dispatch(login(userId)),
    logout: () => dispatch(logout)
  };
};

export const connectUser = connect(
  mapStateToProps,
  mapDispatchToProps
);

export default (cmp) => connectUser(cmp);
```

Code Example

Patterns For Strongly Typed...

- ✅ State Management
- ✅ Redux Connected Components
- Higher Order Components
- Miscellaneous

Higher Order Components

```
const ImagesFetcher = (Component) =>
  class extends React.Component{
    state = { isLoading: false }
    public componentDidMount() {
      if (!this.props.images.length && !this.state.isLoading) {
        this.props.fetchImages("wheredidthesodago");
        this.setState({ isLoading: true })
      }
    }
    public componentDidUpdate(prevProps) {
      if (this.props.images && this.state.isLoading) {
        this.setState({ isLoading: false })
      }
    }
    public render() {
      return <Component {...this.props} isLoading={this.state.isLoading} />;
    }
  };
```

Code Example

Patterns For Strongly Typed...

- ✅ State Management
- ✅ Redux Connected Components
- ✅ Higher Order Components
- Miscellaneous

Miscellaneous

Miscellaneous

1. Typed Webpack
2. Type definition files
3. Untyped dependancies
4. Enums
5. Conditional types
6. Reading long error messages (don't panic)

Additional Resources

- @Piotrwitek's React Redux TypeScript Guide
- React HoC Patterns in TypeScript
- Ultimate React Component Patterns
- A.Sharif's Notes on TypeScript
- TypeScript Deep Dive on Gitbooks
- React Hooks in TypeScript

Thank You!

@marcmissey - Twitter
@marc - Denver Devs