UDC 004.005

APPLYING OF REACT NATIVE FRAMEWORK ON THE EXAMPLE OF DEVELOPING A CROSS-PLATFORM APPLICATION FOR LEARNING ENGLISH WORDS

ANDREI PIROVICH, DMITRY PIATKIN
Polotsk State University, Belarus

This article discusses technologies for implementing a cross-platform application for learning English words.

Free communication is currently one of the main factors in the success of modern man. And the main language is certainly English. English for communication between people and various countries for whom this language is not native. Thus, English has become a necessary attribute of a business person. And my application will help to replenish your vocabulary at any convenient time: in the metro, in a break between work or household chores. And in order to make a decision for more people, I use the cross-platform framework React Native.

React Native is a framework for developing cross-platform iOS and Android applications. It is designed to build a user interface from components, both standard and custom. This framework has the following factors: cross-platform, good performance and an approach to building a UI, which makes it easy to split the interface into independent components.

React Native uses the Flux architecture. Flux is a new architectural approach that complements React and the principle of unidirectional data flow. To implement this approach used the most famous Redux tool.

Features of Redux:

1. Redux uses only one store for the entire application state.
2. Store has read-only access. The only way to change the state is to pass "action" – an object that describes what happened.
3. Changes are made by "clean" functions.

Listing 1.

```
<?
export default (state, action) => {
  switch (action.type) {
    case types.LOAD_VOCABULARY_SUCCESS:
      return action.vocabulary;
      …
    default:
      return state;
  }
}
```

Main parts of Redux:

1. Actions – it are a signals that the store will change. These are clear functions. Action must have a type field. This is the field by which the future will be indexed exactly what happened in our system. They created Action-Created functions.
2. Dispatcher – receives actions on the input and sends these actions (and associated data) to the registered handlers.
3. Stores – containers for the state of applications and business logic in handlers registered in Dispatcher.
4. Controller Views – React-components that collect the state of the repositories and pass it to the child components via props.

Redux offers to keep the entire state of applications in one place, called the "store". Components "Dispatch" the state change to the repository, not directly to the other components. The components that must be aware of these changes are signed to the repository.

Event loop for Redux is shown in figure 1:

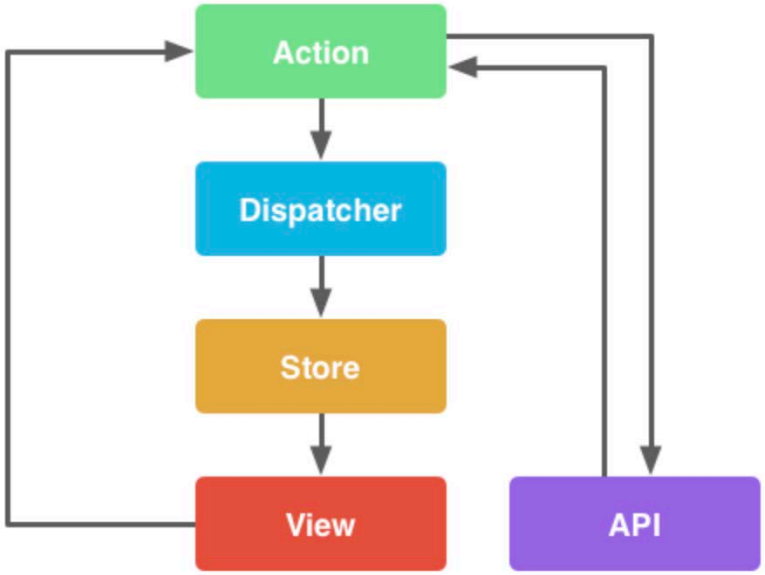

Fig. 1. Functional structure of the application

Developing applications will helpfully understand and implement the React Native technology.

React native updates the tree using Virtual DOM unlike other cross-platform frameworks. It updates only that part of the tree house that requires it. And because of this, the speed of the application increases several times compared to other cross-platform applications.

REFERENCES

1. https://habrahabr.ru/post/246959/. – Access Date: 09.09.2017.
2. https://redux.js.org/. – Access Date: 09.09.2017.
3. https://facebook.github.io/react-native/. – Access Date: 09.09.2017.