Problem a:

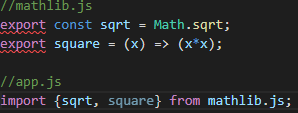
Q: In your own words, explain what a JavaScript module is. What problem is solved by modules? How are modules implemented in JavaScript ES6?

A:

A module in JavaScript is a unit with different functionality to help us organize an app better. It can package functions, classes and components into different namespace without disrupting the whole app at the outer level.

The module solves the problem of maintainability, namespacing and reusability. To be specific, in JavaScript variables outside the functions are global by default. So there is the possibility of namespcing pollution. But the emergence of module has solved the problem.

In ES6 standard, modules can be implemented using Export and Import syntax. The “export” keyword allows you to export symbols from your module to be available to other modules. And the “import” statement allows you to import all or just specific symbols for a module.



As is shown in the picture, in mathlib.js we define two symbols named “sqrt” and “square”. And the symbols can be exported to other files. So in the app.js we import “sqrt” and “square” from mathlib.js. That’s the way modules are implemented in JavaScript ES6.

Program b:

Q: Explain the trade-off between freedom and convention in application organization. Describe the advantages and disadvantages and disadvantages of each approach and give your opinion about which approach you favor.

A:

Freedom:

* Advantages:

The way we organize our code into folders, how we name files and components and how much code we should even put into each file are up to us. So in theory, developers have complete freedom and flexibility to do things we have mentioned however they see fit.

* Disadvantages:

On condition that standards of distinct developers are not the same, developers could not easily support each other's work, which means the maintainability and the adaptability to different platforms is restricted.

Convention:

* Advantages:

First, it’s easy to maintain a cross platform codebase with a minimum of time and cost of supporting multiple target platforms.

Second, developers are able to reuse as much code as possible between two platforms such as iOS and Android.

Third, the convention makes it easy to change configurations of apps without worrying about breaking something else in the code.

* Disadvantages:

Developers have to give up some freedom on adopting conventions.

* My opinion:

In my opinion, I prefer the type of convention. Because as is said in the lecture, while we have given up our freedom of assigning the folders or how to configure our apps, the adaptability and maintainability of our apps are better, which means that our app can also be tested or debugged by other developers following the same standard of convention and that once we have finished our apps on a certain platform, we can transfer our apps using the same codebase into other platforms without too many changes.

Program c:

Q: Define and compare the concepts of:

1. “write once, run everywhere”
2. “learn once, write everywhere”
3. What is React Native considered to the second type of development system?

A:

* “write once, run everywhere”

The definition of “write once, run everywhere” is that apps developed from one device could be expected to run reasonably well on all devices including phones, tablets and desktops regardless of the hardware or operating system.

* “learn once, write everywhere”

There are no agreements between different platforms like iOS and Android, but different platforms share the similar features of practice, idiomatic design and user interface at the higher level. As a result, we can abstract those kinds of similarities to construct our apps. However, because of the differences between different platforms, we need to do some changes in our codes to fit each platform. The concept mentioned above refers to the so-called “learn once, write everywhere.

* React-Native system

React-Native is providing a web like abstraction on top of multiple platforms specifically iOS and Android, and does a good job at making differences between two separate platforms as transparent as possible.

But to fit in different platforms, the react native platform ships with a library of built in components allowing the developer to create functional sophisticated apps without needing to include external third party library components.

Problem d:

Q: Why do you think that React Native needs to provide support for platform-specific components in its standard component library?

A:

As is mentioned in the Problem b, different platforms have their own distinctive features that cannot be transferred to other platforms. Therefore, developers have to choose the platform with specific components to have their app be more authentic and intuitive for users on those target platforms. That is to say React-Native has to provide support for platform-specific components to help developers whose apps need fitting in to different platforms.

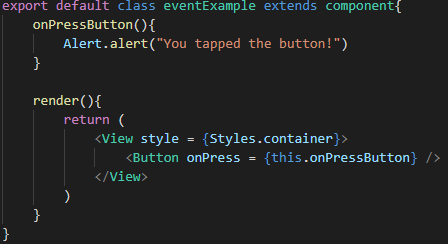
Problem e:

Q: Describe the event system in React Native and explain how to handle events in a React Native App.

A:

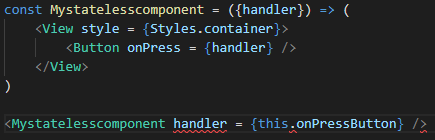
In React-native, events refer to something happens asynchronously regardless of the app workflow. For example, touchable presses, button clicks or timer conflations can be considered as events.

Events in React-Native are handled through handlers registered on components, as known as callback functions. In stateful components, events are handled using a class method as a handler. Then the stateful component returns the value of the callback function to handle the events passed to the callback function. For example:



In this picture, the “eventExample” is a stateful component we define. And the “onPressButton” – the class method – is the callback function. And the “render” function returns to the handler in the end.

On the other hand, in stateless components, it’s hard to handle events. So we need to wrap the stateless component in a stateful component so that the inner components delegate event handling to outer components. For example,



As is shown in the picture, the “Mystatelesscomponent” is a stateless component defined by developers. But as stateless components cannot handle events directly, so we need to wrap it in a stateful component.