# Вопросы. Неделя 26

1. Что такое props и можно ли использовать props в функциональных компонентах?

   Props - объект, данные, которые React передаёт компоненту, чтобы "построить" react-элемент. Да, props можно использовать в функциональных компонентах. В таком случае props будет параметром функции-компонента. То есть Функциональный компонент (как функция), получит данные в одном объекте («пропсе») в качестве параметра и возвратит React-элемент.

2. Нужно ли выделять в отдельный компонент статью в блоге?

   Думаю, да, в том случае, когда статей в блоге много, можно будет составлять статьи с разным наполнением, используя один компонент. Поэтому, скорее да, чем нет.

3. Можно ли использовать React без JSX?

   JSX необязателен для работы с React. Всё, что можно написать и сделать при помощи JSX, может быть сделано на чистом JavaScript.

4. Чем отличается JSX от HTML?

   JSX - это XML-подобный синтаксис. Однако, по своей сути JSX это все-таки, расширение языка JavaScript, и ближе он к JavaScript чем к HTML, например, в том, что JSX React DOM использует стиль именования camelCase для свойств вместо обычных имён HTML-атрибутов.

   Так, class становится className в JSX, а tabindex становится tabIndex.

   А объединяет их то, что одиночные теги в JSX должны быть закрыты: <hr />.

5. Для чего нам нужны свойства (props) компонентов?

   Чтобы наполнять элементы данными, которые передаются в пропсах.

6. В примере с CardList чем можно было бы заменить <React.Fragment>

   Возможно, на тег <ul>, как если бы внутри были элементы списка <li> (Card).

7. Можно ли сказать, что классовые и функциональные компоненты равнозначны по функциональности?

   Можно, после того как в 2019 году команда React в версии 16.8 ввела hooks (хуки), и сделала функциональные компоненты даже более лёгкими в написании, чем классовые.

8. Можно ли полностью описать приложение, используя только функциональные компоненты?

Да.

9. Какой командой мы делаем экспорт компонента для возможности его использования в других местах приложения?

export

10. Изучите структуру компонент в проекте https://github.com/alisa-tsvetkova/EthereumUI и напишите, какой именно компонент является самым верхним, а какой - самым "глубоким"?

Самый верхний, судя по всему, header (есть footer еще, но у него нет вложенных компонентов). В header лежит SearchBar, где инпуты, куда вводится ID, которые валидируются в т.ч. через утилиту helper. В результате проверки выводится BlogViewer, где осуществляется поиск транзакций по ID. Если найдено, запускается компонент BlockInfo – с карточкой с данными найденных транзакций, которые передаются через компонент TranTable (табличка), который и есть самый «глубокий» компонент. Вроде бы так.

11. Какой командой можно сгенерировать разметку/компоненты на основе заранее заданного массива элементов? Приведите пример.

Пример с методом map()

```
/* например, есть компонент Menu, который создает элемент меню бизнес-ланча по
дням недели */
export default App;
import React from "react"

import Menu from "./menu"  //сам компонент меню, который возвращает элемент
import menuData from "./menuData"   //здесь лежит массив - наполнение меню по
дням недели

/* проходим по массиву методом map() и формируем компоненты (экземпляры) с
наполнением из массива */
function App() {
  const menuComponents = menuData.map(menu => <Menu key={menu.weekday}
salad={menu.salad} soup={menu.soup} mainDish={menu.mainDish}
dessert={menu.dessert} />)

    return (
        <div>
            {menuComponents}
        </div>
    )
}

export default App
```