**Питання по React**

1. Що таке React?

React – це інтерфейсна бібліотека JavaScript, розроблена Facebook у 2011 році.

Він слід компонентного підходу і допомагає створювати повторно використовувані компоненти інтерфейсу користувача.

Він використовується для розробки складного та інтерактивного веб-інтерфейсу та мобільного інтерфейсу.

Хоча його вихідний код був відкритий лише у 2015 році, він має велику спільноту підтримки.

1. Які характеристики React?

Основні функції React такі:

* Він використовує віртуальний DOM замість реального DOM.
* Він може виконувати рендеринг на стороні сервера.
* Він слідує односторонньому потоку даних або прив'язці даних.

1. Перерахуйте деякі з основних переваг React.

Деякі з основних переваг React:

* Підвищує продуктивність програми
* можна легко використовувати на клієнті та сервері
* Завдяки JSX код легко читається
* React легко інтегрується з Meteor, Angular та іншими фреймворками.
* Використовуючи React, писати тестові приклади інтерфейсу користувача дуже просто

1. Які обмеження React?

Обмеження реакції наступні:

React - це просто бібліотека, а не повноцінний фреймворк  
Його бібліотека дуже велика, і потрібен час, щоб зрозуміти, що  
початківцям програмістам може бути важко зрозуміти  
, що кодування стає складним, оскільки воно використовує вбудовані шаблони та JSX

1. **Що таке JSX?**

JSX є скороченням від JavaScript XML. Це файл, який використовується React, який використовує виразну потужність JavaScript і синтаксис HTML-подібних шаблонів. Це полегшує розуміння HTML-файлів. Цей файл робить додаток дуже надійним і може поліпшити його продуктивність.

1. **Чи знаєте ви віртуальний DOM? Поясніть, як це працює**

Об'єктна модель віртуального документа (DOM) - це легкий об'єкт JavaScript, який спочатку був лише копією справжнього DOM. Це дерево вузлів, яке розглядає елементи, їх атрибути та вміст як об'єкти та їх атрибути. Функція реваншу React створює дерево вузлів з компонентів React. Потім він оновлює дерево у відповідь на зміни в моделі даних, викликані різними діями, що виконуються користувачем або системою..

Об'єктна модель віртуального документа (DOM) - це легкий об'єкт JavaScript, який спочатку був лише копією справжнього DOM. Це дерево вузлів, яке розглядає елементи, їх атрибути та вміст як об'єкти та їх атрибути. Функція реваншу React створює дерево вузлів з компонентів React. Потім він оновлює дерево у відповідь на зміни в моделі даних, викликані різними діями, що виконуються користувачем або системою.

1. **Чому мій браузер не може читати JSX?**

Браузер може обробляти тільки об'єкти JavaScript, але не може читати JSX в звичайних об'єктах JavaScript. Отже, для того, щоб браузер міг читати JSX, спочатку потрібно використовувати конвертер JSX, такий як Babel, щоб перетворити файл JSX на об'єкт JavaScript, а потім передати його браузеру.

1. **Що таке реквізит?**

Реквізит є коротким для властивостей у React. Вони тільки для читання і повинні залишатися чистими, тобто незмінними. Вони завжди передаються від батьківського до дитячого компонента протягом всієї заявки. Дочірні компоненти ніколи не можуть надсилати реквізити назад до батьківського компонента. Це допомагає підтримувати односторонній потік даних і часто використовується для представлення динамічно згенерованих даних.

1. **Який стан у React? Як він використовується?**

Стан є ядром компонентів React, джерелом даних, і він повинен бути максимально простим. По суті, стан - це об'єкт, який визначає уявлення і поведінку компонента. На відміну від реквізиту, вони модифікуються і створюють динамічні та інтерактивні компоненти. Доступ до них можна отримати через цей.state().

1. **Що таке функції стрілок у React? Як користуватися?**

Функції зі стрілками (=>) — це короткий синтаксис для написання функціональних виразів. Ці функції дозволяють правильно пов'язувати контекст компонентів, оскільки автоматичне зв'язування не можна використовувати за промовчанням у ES6. При використанні функцій вищого порядку дуже корисні функції стрілок.

1. **Які етапи життєвого циклу компонента React?**

Існує три різних етапи життєвого циклу компонентів React:

* Початковий етап рендеринга: Це етап, на якому компонент збирається почати свій життєвий шлях і увійти в DOM.
* Етап оновлення: Після додавання компонента до DOM його можна оновити та повторно відтворити лише тоді, коли властивість або стан зміниться. Це відбувається тільки на цьому етапі.
* Фаза видалення: Це останній етап життєвого циклу компонента, при цьому компонент руйнується і видаляється з DOM.

1. **Детально поясніть методи життєвого циклу компонента реакції.**

Одними з найбільш важливих методів життєвого циклу є:

* componentWillMount () - запускається перед рендерингом як на клієнті, так і на сервері.
* componentDidMount () - працює тільки на клієнті після першого рендеринга.
* componentWillReceiveProps () - Викликається, коли облікові дані отримуються від батьківського класу і перед викликом іншого рендерера.
* shouldComponentUpdate () - повертає true або false залежно від певних умов. Якщо ви хочете оновити компонент, поверніть true, інакше поверніть false. Типове значення повертає false.
* componentWillUpdate () - викликається перед рендерингом в DOM.
* componentDidUpdate () - викликається відразу після рендеринга.
* componentWillUnmount () - викликається після демонтування компонента з DOM. Використовується для очищення пам'яті.

1. **Що таке події в React?**

У React події спрацьовують у відповідь на певні операції, такі як наведення вказівника, клацання та натискання клавіш. Обробка цих подій схожа на обробку подій у елементах DOM. Але є деякі граматичні відмінності, такі як:

Використовуйте номенклатуру від випадку верблюдного регістру , щоб називати події, а не використовувати лише малі літери.  
Подія передається як функція, а не рядок.  
Параметр події містить набір атрибутів, пов'язаних із подією. Кожен тип події містить свої власні властивості та поведінку, доступ до яких можна отримати лише через обробник подій.

1. **Що таке синтетичні події в React?**

Синтетичні події - це об'єкти, які діють як кросбраузерні оболонки навколо власних подій браузера. Вони об'єднують поведінку різних браузерів в один API. Це робиться для того, щоб подія відображала узгоджені властивості в різних браузерах.

1. **Що ви знаєте про React refs?**

Refs є коротким для посилань у React. Це атрибут, який допомагає зберігати посилання на певний елемент або компонент React, і він буде повернутий функцією конфігурації рендеринга компонента. Використовується для позначення певного елемента або компонента, що повертається функцією render(). Вони стануть в нагоді, коли потрібно виконати вимірювання DOM або додати методи до компонентів.

1. **Як створити форму в React**

Форми реакції схожі на HTML-форми. Але в React держава міститься в державній власності компонента і може бути оновлена тільки за допомогою setState(). Таким чином, елементи не можуть безпосередньо оновити свій стан, і їх подання обробляється функціями JavaScript. Ця функція надає повний доступ до даних, введених користувачем у формі.

1. **Що таке компонент високого порядку (HOC)?**

Компоненти високого рівня - це вдосконалений метод повторного використання логіки компонентів і шаблон компонента, отриманий з React. HOC - це настроюваний компонент, який містить інший компонент. Вони можуть приймати будь-яку динаміку, надану субкомпонентами, але не будуть змінювати або копіювати будь-яку поведінку в своїх вхідних компонентах. Ви можете думати про HOC як про «чистий» компонент.

1. **Що таке чистий компонент?**

Чисті компоненти - це найпростіші і швидкі компоненти, які можна записати. Вони можуть замінити будь-який компонент, який має лише render(). Ці компоненти покращують простоту коду і продуктивність програми.

1. **Яке значення ключа в React?**

Ключ використовується для ідентифікації унікального віртуального елемента DOM і відповідних даних, які керують інтерфейсом користувача. Вони допомагають React оптимізувати рендеринг, повторно використовуючи всі поточні елементи в DOM. Ці клавіші повинні бути унікальними числами або рядками. React просто змінює порядок елементів, а не відображає їх знову. Це може підвищити продуктивність вашої програми.

1. **Які основні проблеми фреймворку MVC?**

Нижче наведено деякі з основних проблем з інфраструктурою MVC:

* Вартість маніпулювання DOM дуже висока
* Програма працює повільно і неефективно
* Серйозна трата пам'яті
* Через циклічні залежності складові моделі повинні створюватися на основі моделей і поглядів.

1. **Що таке Redux?**

Redux є однією з найпопулярніших зовнішніх бібліотек розробки на сьогоднішній день. Це передбачуваний державний контейнер для програм JavaScript, який використовується для управління станом всього додатка. Додатки, розроблені за допомогою Redux, легко тестуються, можуть працювати в різних середовищах і проявляти послідовну поведінку.

1. **Яких трьох принципів дотримується Redux?**

Єдине джерело істини: Стан всього додатка зберігається в дереві об'єктів / станів в одному сховищі. Одне дерево стану дозволяє легко відстежувати зміни з плином часу та налагоджувати або тестувати програми.

Статус лише для читання: Єдиний спосіб змінити стан - це почати дію. Дії - це загальні об'єкти JS, які описують зміни. Так само, як стан є найменшим представленням даних, ця операція є найменшим представленням змін даних.

Використовуйте чисті функції для зміни: Щоб вказати, як дерево стану перетворюється операціями, вам потрібні чисті функції. Чисті функції - це ті, чиє значення повернення залежить тільки від значень їх параметрів.

1. **Що ви маєте на увазі під "єдиним джерелом істини"?**

Redux використовує «Магазин» для зберігання всього стану програми в одному місці. Таким чином, статус всіх компонентів зберігається в Магазині, і вони отримують оновлення з самого Магазину. Одне дерево стану дозволяє легко відстежувати зміни з плином часу та налагоджувати або тестувати програми.

1. **Перелічіть компоненти Redux**

Redux складається з наступних компонентів:

* Дія - це об'єкт, який використовується для опису того, що сталося.
* Редуктор - це місце, де можна визначити, як зміниться стан.
* Цех. Дерево стану/об'єкта всієї програми зберігається в Магазині.
* Preview - відображення тільки даних, наданих Магазином.

1. **Як визначити дію в Redux?**

Дія в React повинна мати атрибут типу, який визначає тип дії для виконання. Їх необхідно визначити як струнні константи, а до них можна додати додаткові атрибути. У Redux дії створюються функціями, які називаються Творцями дій.

1. **Поясніть роль Редуктора.**

Редуктори - це чисті функції, які визначають, як змінюється стан програми у відповідь на ДІЮ. Коробки передач працюють, приймаючи попередній стан і дію, а потім повертаючись до нового стану. Він визначає, яке оновлення виконувати, залежно від типу операції, а потім повертає нове значення. Якщо вам не потрібно виконувати завдання, воно повернеться до початкового стану.

1. **Що означає Магазин у Redux?**

агазин - це об'єкт JavaScript, який може зберігати стан програми та надавати методи доступу до стану, планування операцій та реєстрації слухачів. Весь стан/дерево об'єктів застосунку зберігається в одному сховищі. Тому Redux дуже простий і передбачуваний. Ми можемо передавати проміжне програмне забезпечення на склад для обробки даних і запису різних операцій, які змінюють стан зберігання. Всі операції повертають новий стан через редуктор.

1. **Які переваги Redux**

Переваги Redux наступні:

* Передбачуваність результатів - Оскільки завжди є джерело істини, сховище, не виникає питання про те, як синхронізувати поточний стан з діяльністю та іншими частинами програми.
* Ремонтопридатність - код стає простіше в обслуговуванні, з передбачуваними результатами і суворою структурою.  
  Рендеринг на стороні сервера. Вам потрібно лише завантажити сховище, створене на сервері, на клієнт. Це дуже корисно для початкового рендеринга і може оптимізувати продуктивність програми для поліпшення користувацького досвіду.
* Інструменти для розробників. Від операцій до зміни статусу розробники можуть відстежувати все, що відбувається в додатку в режимі реального часу.
* Спільноти та екосистеми. Redux має величезну спільноту за ним, що робить його ще веселішим. Велика спільнота талановитих людей зробила свій внесок у вдосконалення бібліотеки і розробила різні додатки.
* Легко перевірити - Код Redux в основному складається з невеликих, чистих і незалежних функцій. Це робить код перевіреним і незалежним.  
  Organization-Redux пояснює, як саме організований код, роблячи код більш послідовним і простішим у використанні в команді.

1. **Що таке маршрутизація React?**

Маршрутизація React – це потужна бібліотека маршрутизації, побудована поверх React, яка допомагає додавати нові екрани та потоки до програми. Це дозволяє синхронізувати URL-адресу з даними, що відображаються на веб-сторінці. Він відповідає за підтримку стандартизованих структур і поведінки і використовується для розробки односторінкових веб-додатків. Маршрутизація React має простий API.

1. **Навіщо потрібна маршрутизація в React?**

Маршрутизатор використовується для визначення декількох маршрутів.Коли користувач визначає певну URL-адресу, якщо ця URL-адреса відповідає шляху будь-якого "маршруту", визначеного в маршрутизаторі, користувач буде перенаправлений на цей конкретний маршрут. Отже, в основному нам потрібно додати бібліотеку маршрутизатора до нашого додатка, що дозволить нам створити кілька маршрутів, і кожен маршрут надасть нам унікальний вигляд.

1. **Перерахуйте переваги маршрутизатора React**

Так само, як React базується на компонентах, API маршрутизатора React v4 називається "Все про компоненти". Маршрутизатор можна візуалізувати як єдиний кореневий компонент (), в якому ми розміщуємо певні підмарини ().  
Немає необхідності вручну встановлювати історичні значення: у React Router v4 все, що нам потрібно зробити, це обернути маршрутизацію в компонент.  
Пакети розділені: є три пакети: один для Web, Native і Core. Це робить наш додаток більш компактним. На основі подібних стилів кодування легко перемикатися.