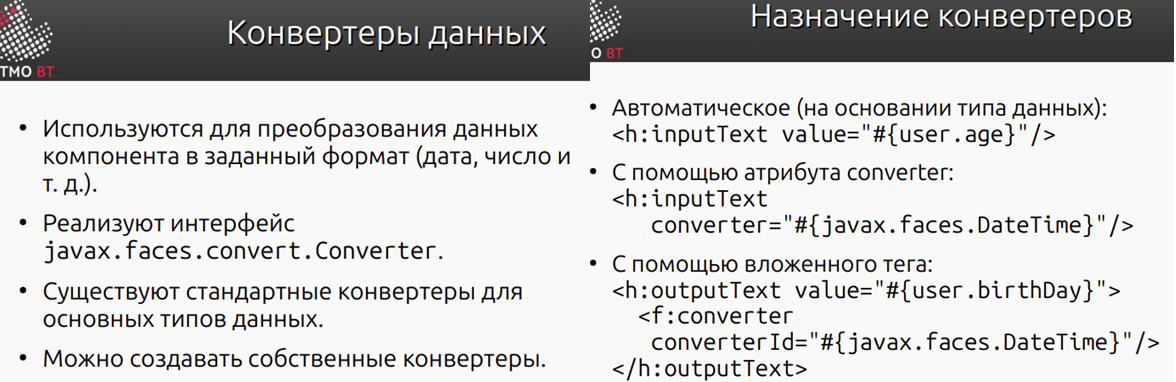
*JSF: Базовые понятия, характеристика технологии, реализуемая модель и прочие общие моменты  
JSF: Managed beans  
JSF: Дерево компонентов и в принципе UI составляющая  
JSF: Конфигурация  
JSF: Жизненный цикл  
JSF: Обработка событий  
JSF: Конверторы и валидторы  
Java EE: общие понятия, общая архитектура  
Java EE: принципы IoC, CDI, Location Transparency  
Java EE: профили платформы  
CDI beans: общее понятие  
CDI beans: аннотации  
CDI beans: именование, "фабрики", "перехватчики"  
ORM и иерархия технологий работы с БД  
JPA: общие принципы  
JPA: конфигурация  
JPA: аннотации  
JAX-RS: общие принципы  
JAX-RS: аннотации  
Frontend: рендеринг  
Frontend: принципы MPA, SPA, PWA, CSR, SSG, SSR  
Frontend: вспомогательные инструменты. Системы сборки. Транспиляция. Package.json. Dev серверы  
React: общие принципы  
React: JSX  
React: компоненты, локальное состояние пропсы  
React: Redux Toolkit  
Управление состоянием (локальное vs глобальное)  
Инструменты глобального управления состоянием и их принципы работы  
Развитие MVC во фронтенде. MVP. MVVM  
Angular: общие понятия, архитектура  
Angular: директивы  
Angular: декораторы  
Angular: компоненты  
Angular: DI*

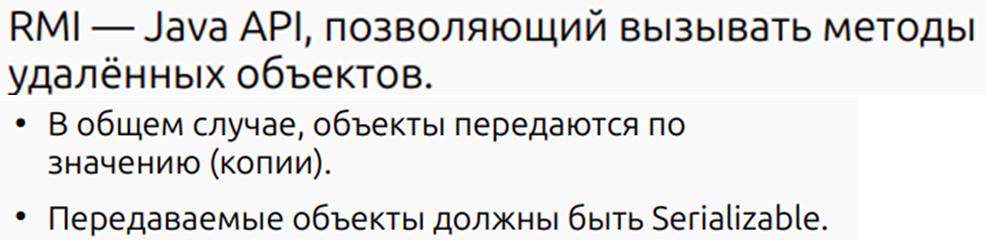
***Билеты:***

Билет 1:

1. Конверторы в JSF



1. RMI в Java EE



**RMI (Remote Method Invocation) в Java** — это **технология распределённых вычислений**, которая позволяет объектам в одной виртуальной машине Java (JVM) вызывать методы объектов, расположенных в другой JVM.

**RMI содержит два основных компонента**:

1. **Клиентский stub**. Взаимодействует с серверным скелетом для вызова удалённых методов и передачи параметров. [1](https://spyboy.blog/2023/06/02/java-rmi-understanding-the-technology-risks-and-best-practices/)
2. **Серверный skeleton**. Получает запрос, вызывает соответствующий метод и возвращает результат. [1](https://spyboy.blog/2023/06/02/java-rmi-understanding-the-technology-risks-and-best-practices/)

**RMI обеспечивает маршалинг данных по сети**

**RMI системы часто включают два отдельных приложения**

1. Написать на реакте формучку для заполнения данных банковской карточки (номер, имя владельца, срок годности и cvv код) (В формочке данных банковской карточке были ограничения, типа номер 16 символов, cvv 3 символа, имя владельца только латинскими и срок годности в формате MM/YY вроде)

const CardForm = () => {

const handleSubmit = (e) => {

e.preventDefault();

const formData = new FormData(e.target);

};

return (

<form onSubmit={handleSubmit}>

<input type="text" name="cardNumber" placeholder="Card Number"

maxLength="16" pattern="[0-9]\*" required />

<input type="text" name="cardHolder" placeholder="Card Holder"

required />

<input type="text" name="expiryDate" placeholder="MM/YY"

pattern="^(0[1-9]|1[0-2])\/\d{2}$" required />

<input type="password" name="cvv" placeholder="CVV"

maxLength="3" pattern="[0-9]\*" required />

<button type="submit">Submit</button>

</form>

);

};

Билет 2:

1. Структура jsf приложения
2. ioc и cdi в javaee
3. Интерфейс на ангуляре, который при наличии сессии показывает формочку для регистрации(Написать на Angular интерфейс, который проверяет если ли в куки sessionid и если нет, отправляет пользователя на аутентификацию по логину и паролю)

Билет 3:

1. Развитие mvc во фронтенде. Mvp
2. react effects что это такое и для чего применяется
3. rest api на jax-rs реализующий сервис бронирования билетов на самолет. сервис должен принимать на вход коды аэропортов вылета и назначение (3 латинских символа) и дату вылета, а на выходе показывать список кодов рейсов (код компании + номер), подходящих под условия

Билет 4:

1. Создание пользовательского интерфейса в jsf приложениях.  
   иерархия компонентов в джсф.
2. Концепция location transparency. реализация в java ee.
3. Приложение на ангуляр, реализующуее форму на отчисление по собственному желанию . должно принимать дату и имя пользователя. и формировать заполненный бланк заваления на клиентской стороне

Билет 5:

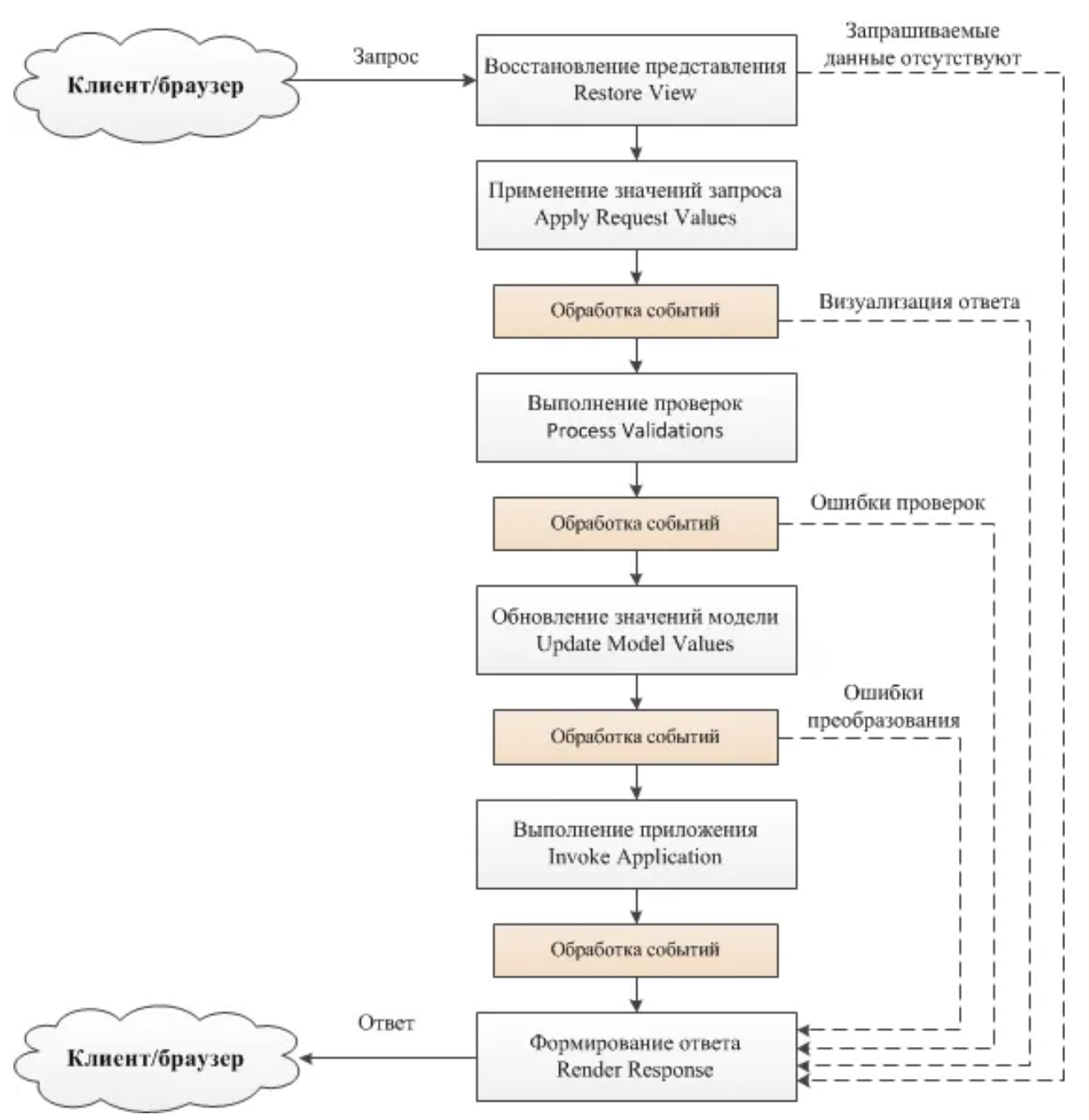
1. Компоненты react
2. Компоненты angular
3. Конфиг jpa entity manager

Билет 6:

1. Управляемые бины
2. JNDI - чё это, способы использования
3. Продемонстрировать слайс в redux toolkit и как его встроить в компонент

Билет 7:

1. Фазы JSF: Invoke Application и Render Response(всё про них расписать).



**Invoke Application**

На данном этапе JSF обрабатывает события, вызванные нажатием кнопок, ссылок и т.д. Присваивает данные модели. Также решаются вопросы навигации(если ее затрагивали).

Вызывается метод UIViewRoot#processApplication() для обработки событий. Перехватить обработку можно переопределением ActionListener

**Render Response**

JSF генерирует ответ, используя данные, полученные на предыдущих шагах. Обновляет представление страницы, используя managed bean, после чего генерируется html страница, которая отправляется пользователю. Так же, если на предыдущих шага происходили ошибки, то они собираются в тег <messages>.

1. Среда выполнения JS. Браузер. Node.js.

Среда выполнения JavaScript — это окружение, в котором выполняется код JavaScript.

Браузеры (Chrome, Firefox, Edge, Safari и др.) содержат встроенные JavaScript-движки (V8, SpiderMonkey, Chakra и др.), которые позволяют выполнять JavaScript-код прямо на веб-странице.

**Ключевые особенности браузерного JS**

✅ **Выполняется в песочнице** – код JS не может получить прямой доступ к файловой системе или операционной системе пользователя.  
✅ **Основная цель – работа с DOM (Document Object Model)** – JavaScript управляет HTML и CSS-элементами на странице.  
✅ **Асинхронность** – поддержка событийного программирования и setTimeout, setInterval, fetch, async/await.  
✅ **Встроенные API браузера** – можно работать с document, window, navigator, fetch, WebSockets, localStorage, sessionStorage, IndexedDB и другими.

Node.js – это среда, которая позволяет выполнять JavaScript вне браузера. Основана на движке **V8** (тот же, что в Chrome).

### ****Ключевые особенности Node.js****

### ✅ **Запуск JavaScript на сервере** – позволяет писать серверные приложения. ✅ **Доступ к файловой системе, процессам, сети** – можно читать/писать файлы, открывать порты и взаимодействовать с операционной системой. ✅ **Использует модульную систему (CommonJS, ES Modules)** – файлы подключаются с require (CommonJS) или import (ESM). ✅ **Поддержка многопоточного асинхронного выполнения** – через Event Loop, Promise, async/await, worker\_threads. ✅ **Менеджер пакетов NPM** – Node.js использует npm для установки библиотек

1. Форма регистрации на React, сервер принимает запрос через REST API. Возвращается страница с формой.

function SignInForm() {

const [login, setLogin] = useState(“”);

const [password, setPassword] = useState(“”);

function signIn() {

const URL = "/api/sign/in"

( async () => {

let data = new FormData();

data.append(“username”, login);

data.append(“password, password);

let result = await fetch(URL, {method: “POST”, body: data});

let message = result.json();

// Логика проверки, успешно ли мы вошли и тд

})()

}

return (

<form onSubmit={signIn}>

<input

placeholder=”login”

onChange={(e) => setLogin(e.target.value)}

/>

<input

placeholder=”password”

onChange={(e) => setPassword(e.target.value)}

/>

</form>

)}

Билет 8:

1. Пакетные менеджеры. Yarn: особенности, основные команды
2. JSX в React
3. Написать с помощью аннотаций конфигурацию CDI бина с именем myBean и каким-то скопом(не помню)

Билет 9:

1. EntityManager в JPA, основные API
2. AJAX в JSF
3. Написать кнопки, позволяющие расшарить в вк, фейсбуке и твиттере

Билет 10:

1. CDI отличия от EJB
2. Директива в ангуляр
3. JSF страница генерящая простые числа

Билет 11:

1. Компоненты React. State и props. Глупые и умные компоненты
2. Компоненты Angular. Чё-то там про реализацию и устройство компонентов
3. Конфига JPA EntityManager...

Билет 12:

1. CDI-bean наименование bean и фабрик bean
2. Транспиляция. Что это такое и для чего.
3. JSF (xhtml и CDI-bean） ввод данные и проверки данные

Билет 13:

1. Интерцепторы. Реализация в Java EE
2. Редукс тулкит отличие от ванильного редукса
3. JSF страничка выводящая новости по клику

Билет 14:

1. managed bean, зачем нужен че такое, способы конфигурации
2. jndi как подключить, че такое зачем надо
3. redux toolkit пример слайса и его использование в комплненте реакт

Билет 15:

1. ajax в jsf
2. jpa: EntityManager основные методы
3. какая то хуйня на ангуляре?

Билет 16:

1. Области видимости бинов
2. ORM какие задачи решают, что такое
3. Написать на ангуляре поле с автодополнением, массив для автодополнения получается с REST API(поисковая строка на англуярке с подсказаками и запросами к беку на ресте)

Билет 17:

1. Плюсы минусы JSF
2. Аннотации jpa
3. Вывод текста постранично в реакте

Билет 18:

1. Зачем нужны пакетные менеджеры. Пример команд с yarn
2. JSX. Синтаксис. Использование в React
3. Написать конфигурацию CDI бина c именем mySampleBean с помощью аннтоаций. Бин должен быть виден во всём приложении  
   @Named(value="mySampleBean")  
   @ApplicationScoped  
   public class MySampleBean(){ // реализацию не просили... }

Билет 19:

1. Аннотации cdm beans
2. MVC модель JST
3. написать компонент для реакта  
   че то такое

Билет 20:

1. FacesServlet - что такое
2. React особенности, +, -
3. Привести пример конфигурации jpa entity manager, управляемого контейнером

Билет 21:

1. Компонентный подход в разработке веб приложения
2. Навигация в Angular. Angular Router
3. Напишите JAX-RS REST для перевода из цельсий в фаренгейты и обратно

Билет 22:

1. Декораторы свойств в ангуляре
2. jpa:способы получения entity manager как конфигурируется
3. Написать rest api на jax rs сервис по отзывам на ФИЛЬМЫ

Билет 23:

1. Основные принципы построения веб приложения (или фронта я точно не помню). И как его связывать с бэком
2. Redux
3. на jax-rs написать crud серв

Билет 24:

1. ксс источники, правила, приоритет
2. MVC - состав, назначение, пример реализации
3. жс функция закрывающая окно браузера если открыто [яндекс.ком](https://xn--d1acpjx3f.xn--j1aef/)

Билет 25:

1. Там entitymanager
2. ORM
3. скрипт на ангуляре по отправке смс в твиттер

То что нашла вне билетов:

1. JAX-RS: общий принцип работы
2. Инструменты уравнения глобальным состоянием
3. Внутреннее устройство и принцип работы Jotai
4. Привести пример исходного кода CDI-бины, реализующего шаблон проектирования «команда» (Command Pattern)
5. RMI/JNDI
6. Какой то блять резолвер прописывать, ТемплейтКонтекст
7. Jakarta EE CDI-Beans: аннотация @Alternative и именование бинов
8. Хуки в React. Что это ? Для чего нужны ?
9. Профили платформы java/jakarta EE