**-------------------------------------------------------------------------------------------------------**

banco de dados-v2.1.pdf

**-------------------------------------------------------------------------------------------------------**

**- Slide 01:**

Interagir com alunos sobre o q entendem por banco de dados

**- Slide 02:**

Interface de acesso (não necessariamente gráfica)

**- Slide 03:**

"Chave primária" - importante para relacionamentos e identificação única

**- Slide 07:**

Instalar o PostgreSQL e *criar o Banco de Dados threeway.*

Algumas operações que iremos executar

Criar tabela cliente:

CREATE TABLE customers (

id SERIAL PRIMARY KEY,

customer\_name VARCHAR NOT NULL,

temperatura INT,

customer\_name VARCHAR NOT NULL

);

*Pegar como base os campos contidos no comando INSERT desse slide*

ALTER TABLE table\_name

ADD COLUMN new\_column\_name data\_type;

**- Slide 08:**

Alterar temporariamente a senha do postges no windows

https://stackoverflow.com/questions/18587710/change-reset-postgresql-user-password-on-windows-7#

*impelemtar sql somente após explicar* ***slide 09***

**- Slide 09:**

inserir registros na tabela cliente

inserir vários registro na tabela livro / ESTOQUE

implementar slide anterior

**- Slide 10:**

implementar

**- Slide 11:**

implementar

**--------------------------------- fim a01 t7 web (1 de 10) --------------------------------------**

**- Slide 15:**

implementar classe Cliente com o método getters / setters

**- Slide 16:**

implementar classe Cliente e Conta

Lembrando que o atributo de nome ‘titular’ é do tipo ‘Cliente’

**- Slide 17:**

implementar classe Agencia

List<Conta> ....

**----------------------------------- fim aula t6 a1 (1 de 8)----------------------------------------**

**- Slide 19:**

Início JDBC

**- Slide 21:**

implementar lab

Adiciona chaves no if, (convenção google)



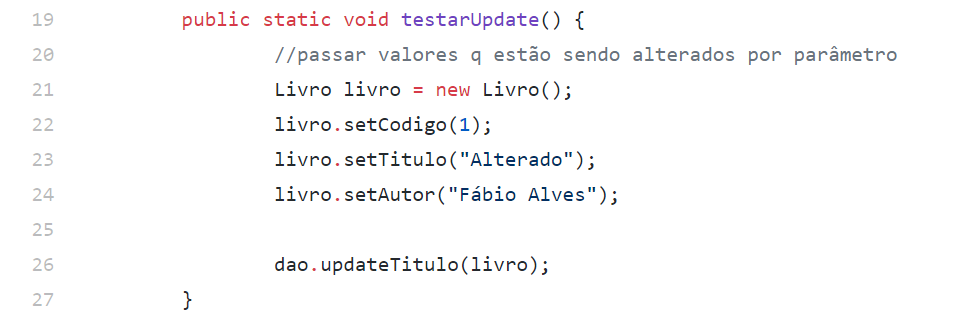
Usar classe exemplo do eclipse

**- Slide 22:**

Executar SQL localizado em: <https://github.com/FabioDevGomes/aulas-3way/blob/master/sql/sqls%20usado%20em%20sala%20aula%20web01%20threeway.sql>

implementar obter (depois de implementar o inserir)

**--------------------------------- fim a02 t7 web (2 de 10) falta testar o inserir com o método main 2 alunos--------------------------------------**



**- Slide 23:**

implementar inserir

**- Slide 24:**

implementar alterar

**- Slide 28:**

implementar salvar

**- Slide 29:**

(Implementado no slide 22)

**- Slide 30:**

(Implementado no slide 24)

**- Slide 31:**

Implementar remover

Ajustar código

- setInt()

- getCodCliente

- Try com recursos

Revisar crud

**-------------------------------------------------------------**

slides fábio.odp

**-------------------------------------------------------------**

What is HTML?

HTML (Hypertext Markup Language) is not a programming language; it is a markup language used to tell your browser

**- slide 3**

acessar **HTML Demo**: <strong>

alterar font-size: 1rem;

https://developer.mozilla.org/en-US/docs/Web/HTML/Element/strong

**- slide 5**

Acessar novamente a página da tag strong e substituir para <b>

**- slide 6**

Implementar

- slide 7

Criar arquivo .css

- slide 10

implementar tag <style> css internal

- slide 11

alterar para css inline

**----------------------------------- Fim aula A3 T6 (3 de 8) ----------------------------------------**

- slide 12

Crie um estilo para o seletor **third** e **second**

- crier uma página HTML e adicione uma imagem

**<img src="dinosaur.png">**

- slide 13

implementar JS inline

alterar para atribuir um texto no lugar da data

- slide 14

implementar JS slide

criar function()

implementar com o texto em português

depois alterar tamanho da fonte

depois ocultar parágrafo

depois, inclua mais campos (<input>) e implemente a validação

- slide 15

testar mudança de estilo via JS (em outra função js)

outra função >> outro botão >> outra tag <p>

no google **import js html**

(JavaScript accepts both double and single quotes)

- slide 16

testar mudança ocultação de elemento via JS (em outra função js)

(JavaScript accepts both double and single quotes)

- slide 17

implementar validação via JS

- slide 18

implementar validação via atributo HTML

- testar api third-party-apis

https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Client-side\_web\_APIs/Third\_party\_APIs

<https://github.com/mdn/learning-area/blob/master/javascript/apis/third-party-apis/youtube/youtube.css>

Implementar exemplo **bootstrap.html**

**baixar bootstrap**

testar:

**<div class="col-sm-6">**

**<div class="container">**

Inclua mais dois campos: **Idade e Profissão** e mostre o valor ao enviar o formulário

----------------------------------------

03\_Servlet\_e\_JSP.pdf

----------------------------------------

- slide 07:

**falar sobre tomcat:**

**mostrar página de download**

**implementar**

- Implementar projeto **servlet**

**http://localhost:8080/servlet/HelloServlet**

*Import Projects from File System or Archive*

Criar segundo servlet em um novo projeto **servlet-jsp**

- servlet-jsp-v1 com hello.jsp

Configurar welcome page **hello.jsp**

- servlet-jsp-v1 com bootstrap.jsp

Ajustar **hello.jsp** para usar bootstrap

<link rel=*"stylesheet"* type=*"text/css"* href=*"css/bootstrap.css"*>

Mostrar como debugar a aplicação

- slide 09:

ajustar implementação anterior para receber parâmetro

criar tela para enviar parâmetro

Exercício: mostrar nome e profissão na tela a partir de um formulário jsp com estilos bootstrap

Saída: Meu nome é **XXXXX** e minha profissão é **XXXXX**

**----------------------------------- Fim aula A4 T6 (de 8) ----------------------------------------**

**Passar GitHub**

- slide 19:

c:forEach

- slide 20: Revisar

Exemplos a implemtentar de JSTL https://www.journaldev.com/2090/jstl-tutorial-jstl-tags-example

Criar **dynamic web project eclipse**

**Criar servlet que contenha uma lista de objetos Livro**



**Definir welcome-file**



**Criar o Servlet**

**Setar listar na requisição** -

request.setAttribute("listaLivrosNoAtributo", listaLivros);

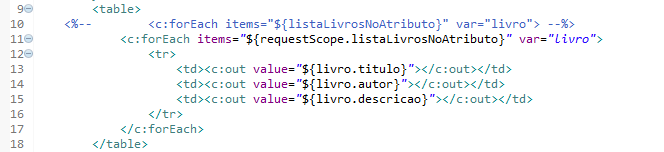
**Redirecionar**:



**Converter projeto para Maven Project para adicionar as dependências**



**Percorrer a lista na JSP usando JSTL:**



(The <c:out> tag can automatically escape XML tags so they aren't evaluated as actual tags.)

**Colorir as linhas:**

<c:forEach items="${requestScope.empList}" var="emp" varStatus="id">

**<tr bgcolor="#${id.count % 2 == 0 ? 'aaee88' : 'ffffff' }" >**

<td><c:out value="${emp.id}"></c:out></td>

<td><c:out value="${emp.name}"></c:out></td>

<td><c:out value="${emp.role}"></c:out></td>

</tr>

</c:forEach>

**Configurar página de erro e passar template usado como exemplo 404.jsp**



**---------------------------------------------------------------------------**

<c:set var="Income" scope="session" value="${4000\*4}"/>

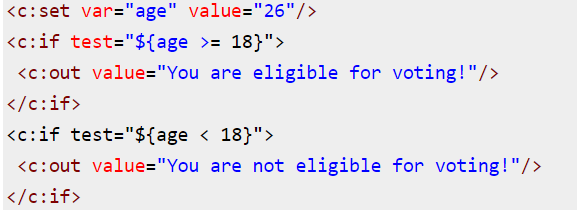
<c:out value="${Income}"/>

<c:remove var="income"/>

<p>After Remove Value is: <c:out value="${income}"/></p>

---------------

**Testar c if**



**Setar mais atributos:**

request.setAttribute("url", "https://www.journaldev.com");

- slide 23:

Passando parâmetros para c:forEach

Projeto exemplo integração front e back-end

------Hello world JSF

http://www.thejavageek.com/2013/06/17/creating-jsf2-hello-world-application-in-eclipse/