

REVISÃO DOS ÚLTIMOS 15 MINUTOS



PERFIL 3

DATAPREV

DESENVOLVIMENTO DE SISTEMAS



@Casa_Bushido

www.casabushido.com.br

© 2026 Casa Bushido. Todos os direitos reservados



**"Torne-se apto em
todas as posições... não
tenha uma postura
favorita."**

Miyamoto Musashi



@Casa_Bushido

www.casabushido.com.br

© 2026 Casa Bushido. Todos os direitos reservados.

Sumário

O cargo	10
Matérias mais importantes	12
Eixo Temático I: Desenvolvimento de sistemas	13
Linguagem de Programação Java	15
Java EE	16
Jakarta EE	17
JPA - Java Persistence API	18
Linguagem de Programação JavaScript	19
JUnit - Testes Automatizados em Java	20
Hibernate - O Framework ORM Poderoso do Ecossistema Java	21
JSF - JavaServer Faces	22
PrimeFaces	23
Desenvolvimento para Dispositivos Móveis	24
Ferramentas Low-Code e No-Code	25
SonarQube	26
Clean Code	27
Arquitetura de Software	28
Interoperabilidade de Sistemas	29
Web Services	30
Mensageria	31
Swagger	32
Arquitetura de Aplicações para Ambiente Web	33
Servidor Web	34
XML	35
XSLT	36
UDDI	37
REST	38

Sumário

DevOps	39
GitLab	40
RPA - Robotic Process Automation	41
Kanban	42
XP - Extreme Programming	43
Padrões de Desenvolvimento e Reuso	44
Codificação de Software - Transacionais, Analíticos, Mobile e API	45
Metodologia de Ponto de Função	46
Story Points	47
Engenharia de Requisitos	48
Técnicas de Elicitação de Requisitos	49
Vue.js	50
Angular	51
Padrões de Frontend	52
Linguagem de Programação React.js	53
SPA e PWA - Aplicações Web Modernas	54
Blockchain	55
Arquitetura Hexagonal	56
Microsserviços	57
Transações Distribuídas	58
User Experience (UX)	59
UX Storytelling	60
Sistemas de Gestão de Conteúdo	61
Arquitetura de Informação	62
Portais Corporativos	63
Workflow	64
Desenho e Planejamento de Interação em Aplicações Web	65
	66

Sumário

Aprendizado de Máquina Supervisionado	67
Aprendizado de Máquina Não Supervisionado	68
IA Generativa	69
Redes Neurais	70
Big Data	71
Eixo Temático II: Testes	72
Teste de Software	74
Testes Unitários	75
Testes de Integração	76
Testes Ágeis	77
Teste de Usabilidade de Software	78
Test-Driven Development (TDD)	79
Testes Automatizados	80
Eixo Temático III: Inteligência de Negócios	81
Business Intelligence (BI)	83
Sistema de Suporte à Decisão	84
Sistema de Gestão de Conteúdo	85
ETL - Extract, Transform, Load	86
OLAP	87
Data Warehouse	88
Modelagens Multidimensionais	89
Business Intelligence	90
Ferramentas de BI	91
Técnicas de Modelagem de Datawarehouse e Datamining	92
BD Individuais e Cubos	93
Técnicas para Coleta de Dados	94

Sumário

Eixo Temático IV: Segurança da informação	95
Políticas de Segurança da Informação	97
ISO/IEC 27001:2022 e 27002:2022	98
ISO/IEC 27002:2022	99
ISO/IEC 27001:2022	100
Gestão de Identidade e Acesso	101
OAuth2	102
SSO - Single Sign-On	103
Planejamento, Identificação e Análise de Riscos	104
ISO/IEC 27005:2019	105
Gestão de Riscos	106
Ameaça, Vulnerabilidade e Impacto	107
OWASP Top 10 2025	108
Ciclo de Vida de Desenvolvimento Seguro (SDL)	109
SAST, DAST e IAST	110
SAST e DAST	111
SAST - Static Application Security Testing	112
DAST - Dynamic Application Security Testing	113

Sumário

Eixo Temático V: Banco de dados	114
Modelagem de Dados - Conceitual, Lógica e Física	116
Normalização de Dados	117
Integridade Referencial	118
Modelagem Dimensional	119
Linguagem de Consulta Estruturada (SQL)	120
Linguagem de Definição de Dados (DDL)	121
Linguagem de Manipulação de Dados (DML)	122
SGBD - Sistema Gerenciador de Banco de Dados	123
NoSQL	124
Banco de Dados em Memória	125
Dado, Informação, Conhecimento e Inteligência	126
Dados Estruturados e Dados Não Estruturados	127
ETL/ELT	128
Transferência de Arquivos e Integração via Base de Dados	129

Sumário

Eixo Temático VI: Gestão e governança de TI	130
PMBOK 7ª Edição - Área de Conhecimento	132
PMBOK 7ª Edição - Grupos de Processos	133
PMBOK 7ª Edição - Processos	134
Projetos, Programas, Portfólio	135
Tipos de Abordagem: Tradicional, Híbrida e Ágil	136
Metodologia Lean	137
Kanban	138
Scrum	139
Gestão de Riscos	140
COBIT 2019 - Governança e Gestão de TI Orientadas a Valor	141
COBIT 2019 - Conceitos Básicos, Estrutura e Objetivos	142
ITIL v4 - Conceitos Básicos, Disciplinas, Estrutura e Objetivos	143
ITIL v4 - Gerenciamento de Serviços	144
Moderno, Ágil e Alinhado a Valor Gestão Ágil de Projetos	145

Apresentação

Olá pessoal! Obrigado por adquirirem o primeiro material da Casa Bushido, esse apoio é muito importante para um futuro brilhante que estou traçando para nós concurseiros de TI.

Esse material é totalmente direcionado para as específicas para o **cargo de Desenvolvimento de sistemas** o coração da TI, estamos de frente com o cargo com mais matérias do concurso, então é muito importante que o estudo já esteja avançado.

A **revisão de 15 minutos** foi baseado nos meus materiais de revisão que juntei ao longo de 8 anos de estudo de concurso. Ele foi feito principalmente para revisar naqueles último minutos que estamos esperando para fazer a prova.

Este material não foi criado para estudo e nem para concurseiros iniciantes. Foi feito para revisão. Apesar de servir como base e também ter os assuntos mais cobrados ele não é para ensinar, é para revisar.

Seja bem vindo. E boa prova!



Com carinho
Vitor Scooby

O cargo

Análise do cargo

Cargo fundamental e mais concorrido do concurso. Os editais de Desenvolvimento de Sistemas geralmente são muito extensos e com cobrança de várias linguagens de programação. O Dataprev é dominado por programadores que são a base para inovação no órgão.

**Vagas
imediatas**

80

Vagas + CR

451

**Peso das específicas
na prova (2,5)**

65,2%

**Total de pontos
das específicas**

75 pt



Atribuições do cargo



Garantir a entrega e manutenção dos **serviços de TIC** dentro dos SLAs e NDAs.



Gerenciamento de **incidentes, problemas, mudanças**, liberações e configuração.



Gestão de **capacidade, disponibilidade e custos** da infraestrutura.



Prestar suporte técnico e administrar **ativos de rede** e infraestrutura de servidores.



Avaliar e manter a esteira de conformidade utilizando normativas e **auditorias internas/externas**.



Realizar **Forense Computacional** subsidiando investigações internas e externas.

Matérias mais importantes

Disparado este é o cargo com mais matérias do concurso, então as questões ficam todas pulverizadas no edital. Claramente Banco de dados e Programação são as matérias mais importantes, mas o edital também cita Teste de software que sempre é questão certa. E Gestão e governança com questões “derruba candidato”

Há também uma certa densidade em DevOps, CI/CD, Nuvem e IA

As matérias mais importantes baseadas nas últimas provas da FGV são:

- Linguagens de programação (em especial Java e frameworks)
- Banco de dados
- Ágil (tanto programação quanto testes)
- Data warehouse e ETL
- OWASP 10



EIXO TEMÁTICO I

DESENVOLVIMENTO DE SISTEMAS



Quais matérias focar

Estamos com o maior edital do concurso e com matérias mais pulverizadas. Isso é comum em cargo de Desenvolvimento de sistemas.

É arriscado apontar a linguagem de programação mais importante, geralmente é Python e Java. Mas não dá para deduzir com tanta força, ainda mais com esse tanto de linguagem de programação.

23%

Desenvolvimento Web e Interoperabilidade

JSON, Web Services e o funcionamento conceitual de Single Page Applications (SPA) além dos frameworks como React, Angular e VueJS estão muito frequentes

21%

Java

IA e Machine Learning

Se encontra nessa parte e será questão certa. Focar em Aprendizado Supervisionado

守護

SHUGO

Escreva uma vez,
rode em qualquer
lugar.



LINGUAGEM DE PROGRAMAÇÃO

JAVA

ROBUSTA. PORTÁTIL. ORIENTADA A OBJETOS. MULTIPLATAFORMA.

Java é uma linguagem de programação de alto nível, criada pela Sun Microsystems (1995) e mantida pela comunidade OpenJDK. É projetada para ser independente de plataforma, segura, escalável e ideal para aplicações corporativas, mobile, web, desktop e sistemas embarcados.

知恵

CHIE

Conhecimento
é a melhor
ferramenta
para criar o
futuro.



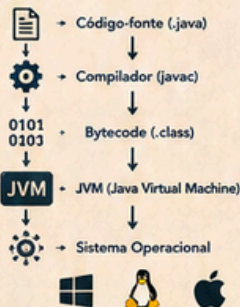
1 O QUE É JAVA?

Java é uma linguagem de programação compilada e interpretada, orientada a objetos e de propósito geral.

- ✓ Simples e familiar
- ✓ Portável e plataforma neutra
- ✓ Segura e confiável
- ✓ Alta performance
- ✓ Escalável e distribuída
- ✓ Multithread e concorrente
- ✓ Robusta e resistente



2 COMO FUNCIONA?



3 ESTRUTURA BÁSICA

```
public class Exemplo {
    public static void main(String[] args) {
        System.out.println("Olá, Mundo!");
    }
}
```

- class: define a classe
- public: nível de acesso
- static: pertence à classe
- void: não retorna valor
- main: ponto de entrada
- System.out.println(): exibe saída



4 PILARES DA OO

- Encapsulamento**
Protege dados e controla o acesso.
- Herança**
Reutiliza e estende comportamentos.
- Polimorfismo**
Objetos diferentes, mesma interface.
- Abstração**
Foca no essencial, esconde os detalhes.

5 ONDE JAVA É UTILIZADA?

- Coleção**
Estruturas de dados poderosas e eficientes.
- Tratamento de Exceções**
Código mais robusto e confiável.
- Stream API**
Processamento funcional de dados.
- Lambda**
Expressões concisas e funcionais.
- Modules (JPMS)**
Modularização da aplicação.
- I/O e NIO**
Entrada/saída rápida e não-bloqueante.
- Garbage Collector**
Gerenciamento automático de memória.

6 ONDE JAVA É UTILIZADA?

- Sistemas Corporativos
- Aplicações Web
- Android (Kotlin/Java)
- Big Data (Hadoop, Spark)
- Cloud (AWS, Azure, GCP)
- IoT e Dispositivos
- Jogos (Minecraft)

JAVA EM NÚMEROS

- 1995**
Lançamento oficial
- +9 MILHÕES**
Desenvolvedores no mundo
- +3 BILHÕES**
Dispositivos rodam Java (estimado)
- TOP 3**
Linguagens mais usadas no mundo

JAVA VS OUTRAS LINGUAGENS

RECURSO	JAVA	PYTHON	C++
Tipo	Compilada + Interpretada	Interpretada	Compilada
Plataforma	Independente (JVM)	Independente	Dependente
Tipagem	Forte e Estática	Dinâmica	Estática
Performance	Alta	Média	Muito Alta
Uso Principal	Corporativo, Mobile, Web	IA, Dados, Scripts	Sistemas, Jogos

CURIOSIDADES

- O nome "Java" vem da ilha de Java, na Indonésia, referência ao café.
- O mascote é o "Duke".
- "Write Once, Run Anywhere" é o lema da linguagem.
- Java foi a primeira linguagem a popularizar a ideia de máquinas virtuais.



FERRAMENTAS POPULARES

- IntelliJ IDEA
- Eclipse
- NetBeans
- VS Code
- Maven

BOAS PRÁTICAS

- ✓ Siga os princípios SOLID
- ✓ Escreva código limpo e legível
- ✓ Trate exceções adequadamente
- ✓ Teste sua aplicação
- ✓ Mantenha dependências atualizadas



PALAVRAS-CHAVE

- JVM
- JDK
- JRE
- OOP
- API
- JDK
- GC
- JAR
- Maven

LEMBRE-SE

Java não é apenas uma linguagem. É uma plataforma completa para criar soluções robustas e escaláveis.



JAVA: UMA LINGUAGEM ATEMPORAL, CONFIÁVEL E EM CONSTANTE EVOLUÇÃO.

改善

KAIZEN

Melhoria contínua, excelência em cada solução.



瀟任

SEIKININ

Responsabilidade e qualidade em cada linha de código.



JAVA EE

PLATAFORMA EMPRESARIAL PARA A CONSTRUÇÃO DE APLICAÇÕES ROBUSTAS, ESCALÁVEIS E SEGURAS

Java EE é um conjunto de especificações e tecnologias que define um padrão para o desenvolvimento de aplicações corporativas na plataforma Java, promovendo produtividade, portabilidade, segurança e integração.



ROBUSTEZ



ESCALABILIDADE



SEGURANÇA



PADRONIZAÇÃO



INTEGRAÇÃO

01 O QUE É JAVA EE?



Plataforma para desenvolvimento de aplicações corporativas distribuídas, baseada na linguagem Java.

- Padronizada pela Jakarta EE (antiga Java EE)
- Suporta aplicações de missão crítica em ambientes robustos

02 ARQUITETURA



Baseada em componentes modulares executados em um servidor de aplicações (Application Server).



Cliente



Servidor de Aplicações



Recursos Corporativos

03 PRINCIPAIS TECNOLOGIAS

- JSF** JavaServer Faces (UI Web)
- JSP** JavaServer Pages
- EJB** Enterprise JavaBeans
- CDI** Contexts and Dependency Injection
- JPA** Java Persistence API
- JTA** Java Transaction API
- JMS** Java Message Service
- JAX-RS** Serviços RESTful
- JAX-WS** Serviços Web SOAP
- JCA** Conectores para sistemas legados
- Bean Validation** Validação de dados
- JAAS / JACC** Segurança e Controle de Acesso

APRESENTAÇÃO

JSF · JSP · Facelets

SEGURANÇA

JAAS · JACC
SSL/TLS

NEGÓCIOS

EJB · CDI
JAX-RS · JAX-WS

MENSAGERIA

JMS · JCA

PERSISTÊNCIA

JPA · JDBC · JTA



04 FUNCIONAMENTO



Os componentes se comunicam através de serviços gerenciados pelo servidor de aplicações, que cuida de:

- ✓ Gerenciamento de ciclo de vida
- ✓ Transações
- ✓ Segurança
- ✓ Concorrência
- ✓ Recursos compartilhados

05 VANTAGENS

- ✓ Alta produtividade
- ✓ Reutilização de componentes
- ✓ Portabilidade (Write Once, Run Anywhere)
- ✓ Segurança integrada
- ✓ Escalabilidade e performance
- ✓ Integração com sistemas corporativos

06 CASOS DE USO

- ✓ Sistemas bancários
- ✓ Portais corporativos
- ✓ E-commerce
- ✓ Soluções governamentais
- ✓ Aplicações de RH e ERP
- ✓ APIs e serviços empresariais

FLUXO DE UMA APLICAÇÃO JAVA EE



Cliente (Navegador/ Aplicação)



Camada de Apresentação (JSF/JSP)



Camada de Negócio (EJB/CDI)



Camada de Persistência (JPA/JDBC)



Banco de Dados / Recursos Externos

CONTAINER DO SERVIDOR DE APLICAÇÕES



WEB CONTAINER



EJB CONTAINER



TRANSACTION MANAGER



SECURITY MANAGER



CONNECTOR CONTAINER



RESOURCE ADAPTERS

PADRÕES E ESPECIFICAÇÕES JAKARTA EE



Jakarta EE Web Profile



Jakarta EE Platform



Jakarta Persistence (JPA)



Jakarta RESTful Web Services



Jakarta Messaging (JMS)

DADO RÁPIDO



85% das grandes organizações utilizam tecnologias Java em suas aplicações corporativas.

*Fonte: New Relic Developer Survey

COMPARATIVO

CARACTERÍSTICA	JAVA EE	SPRING BOOT
Padronização	Alta (especificações)	Convencional
Servidor de Aplicações	Obrigatório	Embutido
Escalabilidade	Alta (ambientes corporativos)	Alta
Integração Empresarial	Nativa	Via bibliotecas
Curva de Aprendizado	Moderada/Alta	Média

BENEFÍCIOS EMPRESARIAIS



Redução de custos operacionais



Mais segurança e conformidade



Implantação confiável



Facilidade de manutenção

BASE LEGAL / REFERÊNCIA



Leis de Software e Direitos Autorais Lei nº 9.609/98

Diretrizes de Segurança da Informação ISO/IEC 27001

JAVA EE: INFRAESTRUTURA SÓLIDA PARA SOLUÇÕES EMPRESARIAIS QUE MOVEM O FUTURO.

改善

KAIZEN

Melhoria contínua e evolução constante.

JAKARTA EE

A PLATAFORMA PADRÃO PARA APLICAÇÕES EMPRESARIAIS MODERNAS EM JAVA

Jakarta EE é a evolução do Java EE, mantida pela Eclipse Foundation. Fornece um conjunto de especificações para construir aplicações corporativas robustas, portáteis, escaláveis e seguras.

PADRÃO ABERTO PORTABILIDADE ESCALABILIDADE SEGURANÇA PRODUTIVIDADE COMUNIDADE

誠実

SEIJITSU

Padrões abertos, transparência e colaboração para um ecossistema mais forte.

01 O QUE É JAKARTA EE?



Conjunto de especificações para o desenvolvimento de aplicações corporativas em Java, executadas em servidores de aplicação.

- ✓ Padrão aberto
- ✓ Independente de fornecedor
- ✓ Compatível com Java SE
- ✓ Ideal para ambientes críticos

02 PRINCIPAIS ESPECIFICAÇÕES

- Jakarta Persistence (JPA)
- Jakarta RESTful Web Services
- Jakarta Enterprise Beans (EJB)
- Jakarta Server Pages (JSP)
- Jakarta Faces (JSF)
- Jakarta CDI (Contexts and Dependency Injection)
- Jakarta Bean Validation
- Jakarta Security
- Jakarta Transactions

03 ARQUITETURA EM CAMADAS

Jakarta EE separa responsabilidades em camadas para facilitar a manutenção, escalabilidade e reutilização.

Camada de Apresentação

Camada de Negócio

Camada de Persistência

Recursos e Serviços

04 FUNCIONAMENTO



As aplicações são executadas em um servidor compatível com Jakarta EE, que fornece serviços essenciais:

- ✓ Gerenciamento de ciclo de vida
- ✓ Transações
- ✓ Segurança
- ✓ Conectividade
- ✓ Mensageria
- ✓ Pool de conexões
- ✓ Injeção de dependências

05 VANTAGENS

- ✓ Alta produtividade
- ✓ Portabilidade entre servidores
- ✓ Escalabilidade horizontal
- ✓ Segurança integrada
- ✓ Integração com tecnologias modernas
- ✓ Suporte da comunidade e da Eclipse Foundation

06 CASOS DE USO

- Sistemas bancários
- Portais corporativos
- E-commerce
- ERPs e CRMs
- Aplicações governamentais
- Soluções de Recursos Humanos

EVOLUÇÃO E ECOSISTEMA



Java EE (1999)



Java EE renomeado para Jakarta EE (2019)



Eclipse Foundation



Comunidade Global Ativa

Sucessora do Java EE, com novo nome, nova identidade e foco em inovação contínua.

FLUXO TÍPICO DE UMA APLICAÇÃO



FIQUE LIGADO!
Jakarta EE 10+ adota o namespace jakarta.* (antes javax.*). Garanta compatibilidade ao migrar suas aplicações!

DADO RÁPIDO

90%

das grandes organizações ainda utilizam aplicações em servidores Java.

*Fonte: New Relic (2024)

COMPARATIVO

CARACTERÍSTICA	JAKARTA EE	SPRING BOOT
Abordagem	Padrão / Especificações	Framework
Servidor	Externo (WildFly, Payara)	Embutido
Configuração	Convencional / Anotações	Convenção sobre configuração
Comunidade	Eclipse Foundation	Comunidade Spring

BENEFÍCIOS

- ✓ Padronização e interoperabilidade
- ✓ Redução de custos
- ✓ Alta disponibilidade
- ✓ Inovação com estabilidade

BASE LEGAL / REFERÊNCIA



Jakarta EE é um padrão aberto mantido pela Eclipse Foundation. Uso livre e sem custos de licenciamento.

TECNOLOGIA, PADRÃO E COLABORAÇÃO PARA CONSTRUIR O FUTURO DAS APLICAÇÕES EMPRESARIAIS.

改善

KAIZEN

Melhoria contínua
leva a sistemas mais simples, confiáveis e sustentáveis.



JPA

JAVA PERSISTENCE API

A ABSTRAÇÃO DA PERSISTÊNCIA DE DADOS NO MUNDO JAVA

JPA é uma especificação da Jakarta EE que define como os objetos Java são mapeados para tabelas de um banco de dados relacional, simplificando o acesso, a manipulação e a consulta de dados.



責任

SEIKININ

Responsabilidade e clareza no código garantem manutenção e evolução seguras.



01 O QUE É JPA?



JPA (Java Persistence API) é uma especificação que padroniza a persistência de dados em aplicações Java, definindo o mapeamento entre classes Java (entidades) e tabelas de banco de dados.

- ✓ Especificação da Jakarta EE
- ✓ Independente do provedor
- ✓ Baseada em padrões (ORM - Object Relational Mapping)

02 COMO FUNCIONA?



A aplicação interage com objetos Java. O JPA Provider cuida de converter essas operações em SQL.

- ✓ Persist (salvar)
- ✓ Merge (atualizar)
- ✓ Remove (excluir)
- ✓ Find (buscar)
- ✓ Query (consultar)

03 PRINCIPAIS ANOTAÇÕES

- @Entity** Marca a classe como entidade
- @Table** Define a tabela no banco
- @Id** Define a chave primária
- @GeneratedValue** Gera valores automaticamente
- @Column** Define colunas
- @OneToOne** Relacionamento 1 para 1
- @OneToMany** Relacionamento 1 para N
- @ManyToOne** Relacionamento N para 1
- @ManyToMany** Relacionamento N para N

ARQUITETURA JPA



04 JPQL - JAVA PERSISTENCE QUERY LANGUAGE



Linguagem de consulta orientada a objetos, independente do banco de dados.

```
SELECT c FROM Cliente c
WHERE c.status = :status
```

Mais legível e portátil que SQL nativo.

05 VANTAGENS



- ✓ Redução de código repetitivo
- ✓ Produtividade elevada
- ✓ Portabilidade entre bancos
- ✓ Manutenção facilitada
- ✓ Integração com Jakarta EE
- ✓ Suporte a transações e cache

06 PROVEDORES JPA



- Hibernate (mais utilizado)
- EclipseLink
- Apache OpenJPA
- DataNucleus

EXEMPLO DE MAPEAMENTO

ENTIDADE JAVA

```
@Entity
@Table(name="cliente")
public class Cliente {
    @Id
    @GeneratedValue
    private Long id;

    @Column(name="nome")
    private String nome;

    @Column(name="email")
    private String email;
}
```



TABELA RELACIONAL

cliente		
id (PK)	nome	email
1	João Silva	joao@email.com
2	Maria Souza	maria@email.com
3	Carlos Lima	carlos@email.com

CURIOSIDADES

- JPA foi criado em 2006 como parte do Java EE 5.
- Desde 2020, faz parte da Jakarta EE.
- Promove desacoplamento entre a aplicação e o banco de dados.

DADO RÁPIDO



da lógica de persistência em aplicações Java empresariais usa JPA ou um de seus provedores.

(Fonte: New Relic Developer Survey 2024)

COMPARATIVO

	JPA (ORM)	JDBC (NATIVO)
Abstração		Baixa
Produtividade		Menor
Portabilidade		Baixa
Manutenção		Mais difícil
SQL	Gerado automaticamente	Escrito manualmente

BENEFÍCIOS

- Desenvolvimento mais rápido
- Menos código boilerplate
- Mais foco na regra de negócio
- Integração nativa com Jakarta EE

BASE LEGAL / REFERÊNCIA



Especificação Jakarta Persistence (anteriormente Java Persistence API)
Parte da Jakarta EE
(<https://jakarta.ee/specifications/persistence/>)

COM JPA, O DESENVOLVEDOR Foca NO QUE IMPORTA:
O NEGÓCIO. O RESTO, O FRAMEWORK RESOLVE.

守護

SHUGO

Escreva código
que transforma
ideias em impacto.



LINGUAGEM DE PROGRAMAÇÃO

JAVASCRIPT

UMA LINGUAGEM. INFINITAS POSSIBILIDADES.

JavaScript é a linguagem da web. Versátil, dinâmica e essencial, permite criar experiências interativas, rápidas e inteligentes no navegador, no servidor e em qualquer dispositivo.

知恵

CHIE

Conhecimento
é a melhor
ferramenta
para criar
o futuro.



1 O QUE É JAVASCRIPT?

JavaScript (JS) é uma linguagem de programação de alto nível, criada para tornar páginas web interativas.

- ✓ Criada em 1995 por Brendan Eich
- ✓ Padrão ECMAScript (ES)
- ✓ Multi-paradigma: imperativa, funcional e orientada a objetos
- ✓ Essencial para Front-end e Back-end

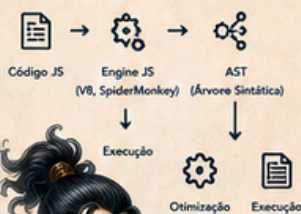


2 CARACTERÍSTICAS PRINCIPAIS

- + Leve e rápida
- + Interpretada
- + Multiparadigma
- + Case-sensitive
- + Tipagem dinâmica
- + Execução única (single-thread)
- + Grande comunidade

3 COMO FUNCIONA?

FLUXO DE EXECUÇÃO



コードは力なり

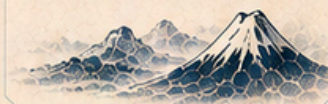


4 SINTAXE BÁSICA

```
// Variáveis
let nome = "Samurai";
const idade = 25;

// Função
function saudacao(nome) {
  return `Olá, ${nome}!`;
}
console.log(saudacao(nome));

// Arrow Function
const soma = (a, b) => a + b;
```



5 TIPOS DE DADOS

- string "texto"
- number 42, 3.14
- boolean true / false
- null null
- undefined undefined
- object { chave: "valor" }
- array [1, 2, 3]
- symbol Symbol('id')

6 APLICAÇÕES

- Desenvolvimento Web (Front-end)
- Back-end (Node.js)
- Aplicações Mobile (React Native)
- Desktop (Electron)
- IoT e Automação
- Jogos e Animações



CURIOSIDADES

- JavaScript foi criado em apenas 10 dias.
- O nome "JavaScript" foi uma estratégia de marketing.
- O símbolo {} vem da linguagem C.
- JS roda em mais de 99% dos navegadores do mundo.



ECOSISTEMA MODERNO



CONCEITOS IMPORTANTES



JAVASCRIPT VS OUTRAS LINGUAGENS

ASPECTO	JAVASCRIPT	PYTHON	JAVA
Execução	Interpretada (Engine JS)	Interpretada	Compilada (JVM)
Tipagem	Dinâmica e fraca	Dinâmica e forte	Estática e forte
Paradigma	Multi-paradigma	Multi-paradigma	Orientada a Objetos
Uso principal	Web, Mobile, Servidores	Ciência de dados, IA	Sistemas corporativos
Desempenho	Alto (otimizações JIT)	Médio	Alto

JAVASCRIPT EM NÚMEROS

- 1995 Ano de criação
- +14M Desenvolvedores no mundo
- +1.8M Pacotes no npm
- Top 1 Linguagem mais usada no Front-end

FERRAMENTAS POPULARES



BOAS PRÁTICAS

- ✓ Use const sempre que possível.
- ✓ Mantenha funções pequenas e focadas.
- ✓ Prefira async/await a callbacks.
- ✓ Escreva código limpo e legível.
- ✓ Teste seu código regularmente.

ESCREVA ONCE. RODE EM QUALQUER LUGAR.

改善

KAIZEN
Melhoria
contínua
através de
testes
confiáveis.

JUnit

TESTES AUTOMATIZADOS EM JAVA

JUnit é o framework de testes unitários mais utilizado no ecossistema Java. Ele facilita a criação, execução e manutenção de testes, garantindo qualidade, confiança e desenvolvimento ágil.



AUTOMATIZA
testes



VALIDA
comportamentos



ENCONTRA
defeitos cedo



FACILITA
refatorações



AUMENTA
a qualidade

品質

HINSHITSU
Qualidade é
construída
em cada
pequeno
teste.

01 O QUE É JUNIT?



Framework open source
para testes unitários
em aplicações Java.
Baseado em anotações,
executa testes de forma
simples e eficiente.

02 PRINCIPAIS ANOTAÇÕES

@Test Marca um método de teste
@BeforeEach Executa antes de cada teste
@AfterEach Executa após cada teste
@BeforeAll Executa uma vez antes de todos
@AfterAll Executa uma vez após todos
@Disabled Desabilita um teste
@DisplayName Define um nome legível
@ParameterizedTest Testes parametrizados

03 ASSERTIONS (AFIRMAÇÕES)

Verificam se o resultado
obtido é o esperado.

assertEquals(expected, actual)
assertTrue(condition)
assertFalse(condition)
assertNull(object)
assertNotNull(object)
assertThrows(Exception.class, () -> {})

CICLO DE VIDA DE UM TESTE



Preparação
(@BeforeEach)



Execução do Teste
(@Test)



Validação
(Assertions)



Limpeza
(@AfterEach)

RESULTADOS POSSÍVEIS



SUCESSO
Teste passou



FALHA
Resultado diferente
do esperado



ERRO
Exceção inesperada
ou problema no teste

EXEMPLO DE TESTE

```
import org.junit.jupiter.api.*;
import static org.junit.jupiter.api.Assertions.*;

class CalculadoraTest {
    private Calculadora calc;

    @BeforeEach
    void setup() {
        calc = new Calculadora();
    }

    @Test
    @DisplayName("Deve somar dois números")
    void deveSomar() {
        int resultado = calc.somar(2, 3);
        assertEquals(5, resultado);
    }
}
```

RELATÓRIO DE EXECUÇÃO

Testes executados: 3 ✓ 2 ✗ 1 0

CalculadoraTest

✓ deveSomar() PASSOU 0.01s
✓ deveSubtrair() PASSOU 0.01s
✗ deveDividirPorZero() FALHOU 0.00s



CURIOSIDADES

- JUnit foi criado por Erich Gamma e Kent Beck em 1997.
- Em 2017, evoluiu para JUnit 5, mais modular e extensível.
- Segue os princípios do Test-Driven Development (TDD).
- Mais de 1 bilhão de downloads por ano no Maven Central.

DADO RÁPIDO



90% dos projetos Java no GitHub usam JUnit para testes unitários.

*Fonte: GitHub Octoverse 2023

COMPARATIVO

RECURSO	JUNIT 4	JUNIT 5
Anotações básicas	✓	✓
Testes parametrizados	(-)	✓
Extensível (Extensions)	(-)	✓
Assinaturas modernas	(-)	✓
Plataforma unificada	(-)	✓

NÍVEIS DE TESTE

Testes de Aceitação (UI)
Testes de Integração
TESTES UNITÁRIOS
(Com JUnit)

BOAS PRÁTICAS

- ✓ Teste um comportamento por método
- ✓ Mantenha testes independentes
- ✓ Use nomes claros e descritivos
- ✓ Siga o padrão AAA (Arrange, Act, Assert)
- ✓ Mantenha os testes rápidos e confiáveis

BASE LEGAL / REFERÊNCIA



Boas práticas de teste apoiam a qualidade de software e conformidade com normas como ISO/IEC 25010.

TESTES CONFIÁVEIS, SOFTWARE DE QUALIDADE.
JUNIT: PEQUENOS TESTES, GRANDES RESULTADOS.

改善

KAIZEN

Melhoria contínua através de boas práticas e evolução constante.



HIBERNATE

O FRAMEWORK ORM PODEROSO DO ECOSISTEMA JAVA

Hibernate é um framework de mapeamento objeto-relacional (ORM) que facilita a persistência de dados em aplicações Java, transformando objetos em registros de banco de dados de forma automática, eficiente e produtiva.



持続

JIZOKU

Persistência confiável, performática e escalável para sistemas robustos.



01 O QUE É HIBERNATE?



Hibernate é um framework ORM para Java que mapeia classes Java para tabelas do banco de dados usando metadados XML ou anotações.

- ✓ Implementa o padrão JPA
- ✓ Reduz código repetitivo (JDBC)
- ✓ Facilita consultas e transações
- ✓ Independente de banco de dados

02 COMO FUNCIONA?

O Hibernate faz a ponte entre sua aplicação Java e o banco de dados.

- Carrega configuração e mapeamentos
- Cria SessionFactory (uma vez)
- Abre Session (conexão lógica)
- Executa operações (CRUD, HQL/JPQL)
- Gerencia transações e flush
- Fecha Session e libera recursos

03 ESTADOS DE UM OBJETO

Os objetos passam por diferentes estados no ciclo de vida gerenciado pelo Hibernate.



04 CONSULTAS

Hibernate suporta diferentes tipos de consultas para recuperar dados de forma flexível.

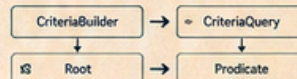
HQL / JPQL

Linguagem orientada a objetos.

```
from Cliente c where c.ativo = true
```

Criteria API

Construção de consultas tipadas.



SQL Nativo

Consultas SQL diretas quando necessário.

```
SELECT * FROM clientes WHERE ativo = 1
```

05 VANTAGENS

- ✓ Acelera o desenvolvimento
- ✓ Menos código repetitivo
- ✓ Independência de banco de dados
- ✓ Cache inteligente (1º e 2º nível)
- ✓ Gerenciamento de transações
- ✓ Integração com Spring e Jakarta EE
- ✓ Suporte a lazy loading e fetch plans

06 RECURSOS AVANÇADOS

- Cache de 1º e 2º nível
- Lazy Loading e Fetch Strategies
- Auditoria e Versionamento (@Version)
- Batch Processing
- Gerenciamento de conexões
- Interceptors e Listeners
- Envers (auditoria de entidades)

CURIOSIDADES

- Criado por Gavin King em 2001.
- Inspirado em EJB Entity Beans.
- Base de muitos frameworks como Spring Data JPA.
- Suporta JPA desde a versão 3.0.
- Comunidade ativa e ampla documentação.

HIBERNATE

OBJECT RELATIONAL MAPPING
ORM

ARQUITETURA HIBERNATE



EXEMPLO DE ENTIDADE E MAPEAMENTO (ANOTAÇÃO)

```
@Entity
@Table(name = "clientes")
public class Cliente {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(name = "nome", nullable = false)
    private String nome;

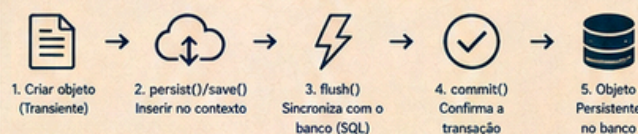
    @Column(name = "email", unique = true)
    private String email;

    // getters e setters
}
```

TABELA GERADA

clientes		
id (PK)	nome	email (UNIQUE)
1	João Silva	joao@email.com
2	Maria Souza	maria@email.com
3	Carlos Lima	carlos@email.com

FLUXO TÍPICO DE PERSISTÊNCIA



DADO RÁPIDO



*Fonte: JetBrains Developer Survey 2023)

COMPARATIVO

RECURSO	JDBC	HIBERNATE
Mapeamento ORM	X	✓
SQL Manual	✓	X
Produtividade	Baixa	Alta
Manutenção	Complexa	Simples
Performance	Média	Alta*

*Com cache e boas práticas

SISTEMAS QUE UTILIZAM



BASE LEGAL / REFERÊNCIA

Especificação Jakarta Persistence (JPA)
EE 10+
Documentação Oficial:
<https://hibernate.org/orm/>

HIBERNATE: MENOS CÓDIGO, MAIS PRODUTIVIDADE, PERSISTÊNCIA INTELIGENTE PARA APLICAÇÕES JAVA.

改善

KAIZEN

Melhoria contínua através de boas práticas e evolução constante.



JSF

JAVASERVER FACES

A TECNOLOGIA DE INTERFACE DA PLATAFORMA JAVA

JSF é um framework para a construção de interfaces de usuário web robustas e reutilizáveis, baseado em componentes, gerenciado por eventos e independente de tecnologias de apresentação.



COMPONENTES REUTILIZÁVEIS



GERENCIADO POR EVENTOS



INTEGRAÇÃO COM JAVA EE



INDEPENDENTE DE TECNOLOGIA



PRODUTIVIDADE E PADRÃO



ROBUSTEZ E SEGURANÇA

調和

CHŌWA

Harmonia entre componentes, camadas e padrões para criar aplicações sólidas e escaláveis.



01 O QUE É JSF?



JSF (JavaServer Faces) é um framework MVC para construção de interfaces web. Ele simplifica o desenvolvimento de aplicações web com componentes de UI, validação, conversão, gerenciamento de estado e navegação.

- ✓ Parte da especificação Java EE
- ✓ Baseado em componentes
- ✓ Independente de tecnologia
- ✓ Fácil integração com outras APIs

02 ARQUITETURA MVC

JSF segue o padrão MVC (Model-View-Controller).



MODEL Beans gerenciados (Managed Beans)



VIEW Páginas Facelets (.xhtml)



CONTROLLER Gerencia eventos e navegação

03 COMPONENTES

JSF fornece um rico conjunto de componentes de interface.

- ✓ Input: `h:inputText`, `h:inputNumber`...
- ✓ Output: `h:outputText`, `h:dataTable`...
- ✓ Seleção: `h:selectOneMenu`, `h:selectMany`...
- ✓ Layout: `h:panelGrid`, `h:panelGroup`...
- ✓ Ações: `h:commandButton`, `h:commandLink`...
- ✓ Validação: `h:message`, `h:messages`...
- + Biblioteca: PrimeFaces, BootsFaces, RichFaces e outras.

04 FUNCIONALIDADES



Gerenciamento de Estado
Mantém o estado da interface entre requisições.



Validação e Conversão
Valida e converte dados de forma declarativa e personalizada.



Internacionalização (i18n)
Suporte nativo a múltiplos idiomas.



Navegação
Navegação declarativa com faces-config ou anotações.



Temas e Estilização
Separação entre estrutura e apresentação.

05 VANTAGENS

- ✓ Acelera o desenvolvimento
- ✓ Padronização e boas práticas
- ✓ Componentes prontos e extensíveis
- ✓ Integração com CDI, JPA, Bean Validation, etc.
- ✓ Suporte a AJAX
- ✓ Independência de tecnologia (JSP, Facelets, HTML5...)

06 INTEGRAÇÃO

- ✓ CDI (Contexts and Dependency Injection)
- ✓ JPA (Persistência de Dados)
- ✓ Bean Validation (JSR 380)
- ✓ Servlet / Filters
- ✓ EJB / JTA
- ✓ WebSockets, REST, JMS, etc.

CURIOSIDADES

- JSF foi criado pela Sun Microsystems e padronizado no JSR 252.
- A partir do Java EE 7 (e Jakarta EE), o JSP não é mais obrigatório, sendo o Facelets o padrão.
- JSF 2.3 trouxe melhorias em AJAX, CDI, WebSockets e componentes.

CICLO DE VIDA JSF



EXEMPLO DE PÁGINA FACELETS (XHTML)

```

<h:form>
<h:panelGrid columns="2">
<h:outputLabel for="nome" value="Nome:" />
<inputText id="nome" value="#{usuarioBean.nome}" />
<h:outputLabel for="salvar" value="Salvar:" />
<commandButton id="salvar" value="Salvar" action="#{usuarioBean.salvar}" />
<messages globalOnly="true" styleClass="erro"/>
</h:panelGrid>
</h:form>
    
```

EXEMPLO DE BEAN GERENCIADO

```

@Named
@ViewScoped
public class UsuarioBean implements Serializable {
    private String nome;

    public void salvar() {
        // lógica de negócio
    }
    // getters e setters
}
    
```

DADO RÁPIDO



das empresas que usam Java EE utilizam JSF para construção de interfaces web.

*Fonte: New Relic Developer Survey 2023

COMPARATIVO

RECURSO	JSF	JSP	Spring MVC
Componentes de UI	✓	✗	✗
Gerenciamento de Estado	✓	✗	✗
Validação Integrada	✓	✓	✗
Navegação Declarativa	✓	✗	✗
Curva de Aprendizagem	Média	Baixa	Média

VERSÕES



BOAS PRÁTICAS

- ✓ Use escopo adequado para beans
- ✓ Prefira Facelets e componentes nativos
- ✓ Valide dados sempre
- ✓ Evite lógica na View
- ✓ Utilize AJAX com moderação

BASE LEGAL / REFERÊNCIA



Especificação Jakarta Faces (antigo JavaServer Faces)
Jakarta EE:
<https://jakarta.ee/specifications/faces>

JSF: COMPONENTES, PADRÕES E PRODUTIVIDADE PARA CONSTRUIR INTERFACES WEB DE ALTO NÍVEL.

改善

KAIZEN
Melhoria
contínua
através de
tecnologia e
inovação.

PRIMEFACES

A BIBLIOTECA DE COMPONENTES JSF
MAIS COMPLETA E PODEROSA

PrimeFaces é uma biblioteca open source de componentes visuais para aplicações JavaServer Faces (JSF), que acelera o desenvolvimento de interfaces web ricas, responsivas e modernas com alta produtividade.

- PRODUTIVIDADE ELEVADA
- COMPONENTES RIQUEZ
- INTEGRAÇÃO COM JSF
- RESPONSIVO E MODERNO
- COMUNIDADE ATIVA
- OPEN SOURCE e GRATUITO

調和

CHŌWA
Harmonia entre
pessoas,
processos e
tecnologia
para entregar
valor.

01 O QUE É PRIMEFACES?



PrimeFaces é uma biblioteca de componentes UI para JSF que fornece um conjunto completo de componentes ricos, temas prontos e utilitários para facilitar o desenvolvimento web.

- Baseado em Java EE / Jakarta EE
- Compatível com JSF 2.x / 3.x
- Fácil integração com CDI, JPA, Bean Validation e outros
- Suporte a AJAX nativo

02 PRINCIPAIS RECURSOS

- Mais de 100 componentes prontos
- Ajax nativo e assíncrono
- Temas responsivos e personalizáveis
- Validação de formulários avançada
- Suporte a Mobile e PWA
- Internacionalização (i18n)
- Acessibilidade (a11y)
- Documentação completa

03 COMPONENTES EM DESTAQUE

- DataTable: Tabelas dinâmicas e avançadas
- Dialog: Janelas modais ricas
- Chart: Gráficos interativos
- FileUpload: Upload com barra de progresso
- Tree: Estruturas hierárquicas
- Galleria: Galerias de imagens
- Calendar: Calendário avançado
- AutoComplete: Sugestão inteligente

04 COMO FUNCIONA?

- Requisição AJAX ou ação do usuário
- JSF + PrimeFaces Processa o evento
- Atualização parcial (Partial Page Update)
- Resposta renderizada somente no necessário

05 VANTAGENS

- Desenvolvimento mais rápido
- Interfaces modernas e elegantes
- Menos código, mais produtividade
- Componentes testados e estáveis
- Grande comunidade e suporte
- Atualizações frequentes

06 INTEGRAÇÃO PERFEITA

- CDI (Contexts and DI)
- Bean Validation
- JPA / Hibernate
- Spring Framework
- Security (JAAS / Spring Security)
- REST Services
- WebSocket

ARQUITETURA DE UMA APLICAÇÃO JSF + PRIMEFACES



CICLO DE VIDA JSF COM PRIMEFACES (AJAX)



EXEMPLO DE CÓDIGO

```
<p:dataTable value="#{clienteBean.lista}" var="c"
  paginator="true" rows="10" responsiveLayout="scroll">
  <p:column headerText="ID" sortBy="#{c.id}">
    <h:outputText value="#{c.id}" />
  </p:column>
  <p:column headerText="Nome" sortBy="#{c.nome}">
    <h:outputText value="#{c.nome}" />
  </p:column>
  <p:column headerText="Ações">
    <p:commandButton value="Editar" icon="pi pi-pencil"
      update="#{form:dig}" oncomplete="PF('dig').show()" />
  </p:column>
</p:dataTable>
```

TEMAS POPULARES



CURIOSIDADES

- Criado por Henrique Dutra em 2008.
- Um dos projetos JSF mais populares do mundo.
- Utilizado por empresas como Mercedes-Benz, Red Hat, Itaú, Embratel e muitas outras.
- Suporte a versões LTS e compatibilidade contínua com Jakarta EE.

DADO RÁPIDO



*Fonte: StackOverflow Survey 2023

COMPARATIVO

RECURSO	JSF PURO	PRIMEFACES
Componentes prontos	Limitados	+100 componentes
AJAX nativo	×	✓
Temas responsivos	Limitados	✓
Gráficos	×	✓
Tabelas avançadas	Básicas	✓
Upload com progresso	×	✓

NÚMEROS DO PROJETO



BOAS PRÁTICAS

- Use componentes nativos antes de criar customizados
- Aproveite o AJAX para updates parciais
- Utilize @ViewScoped para melhor performance
- Valide dados no cliente e no servidor
- Mantenha componentes atualizados

BASE LEGAL / REFERÊNCIA

- Projeto open source sob licença Apache 2.0
- Documentação Oficial: <https://primefaces.org/>

INTERFACES RICAS, ELEGANTES E INTELIGENTES
PARA APLICAÇÕES JAVA COM JSF E PRIMEFACES.

DESENVOLVIMENTO PARA DISPOSITIVOS MÓVEIS

Crie experiências incríveis na palma da mão

- Performance
- Usabilidade
- Segurança
- Conectividade
- Escalabilidade

01 O QUE É?

É o processo de criar aplicativos para smartphones e tablets, explorando recursos do dispositivo e entregando valor ao usuário de forma prática, rápida e segura.

- ✓ Foco em mobilidade
- ✓ Interfaces responsivas
- ✓ Experiência do usuário
- ✓ Integração com hardware
- ✓ Disponível offline/online



02 PRINCIPAIS PLATAFORMAS

ANDROID

- ✓ Linguagem: Kotlin/Java
- ✓ IDE: Android Studio
- ✓ Google Play Store

iOS

- ✓ Linguagem: Swift
- ✓ IDE: Xcode
- ✓ App Store

03 ARQUITETURA TÍPICA



CICLO DE DESENVOLVIMENTO



04 TECNOLOGIAS & FRAMEWORKS

NATIVAS

- Kotlin** (Android)
- Swift** (iOS)

CROSS-PLATFORM

- Flutter** (Dart)
- React Native** (JS)
- Xamarin** (.NET)

HÍBRIDAS / WEBVIEW

- Ionic** • **Cordova** • **Capacitor**

05 RECURSOS MAIS USADOS

- Câmera / Galeria
- GPS / Localização
- Notificações Push
- Biometria / Segurança
- Armazenamento Local
- APIs REST / GraphQL



06 BOAS PRÁTICAS

- ✓ Mantenha a UI simples e objetiva
- ✓ Otimize consumo de bateria e dados
- ✓ Teste em diferentes dispositivos
- ✓ Cuide da acessibilidade
- ✓ Proteja os dados do usuário



CURIOSIDADES

- O mercado mobile cresce mais de 20% ao ano.
- Mais de 6 bilhões de usuários de smartphones no mundo.
- Apps nativos têm melhor performance, mas cross-platform ganha velocidade e economia.



EXEMPLO DE CÓDIGO (Flutter / Dart)

```
import 'package:flutter/material.dart';

void main() => runApp(MeuApp());

class MeuApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text('Olá!')),
        body: Center(child: Text('Bem-vindo!')),
      ),
    );
  }
}
```



DADO RÁPIDO

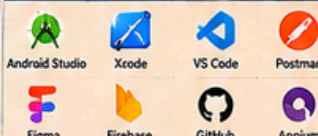


*Fonte: Data.ai 2024

COMPARATIVO RÁPIDO

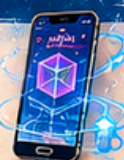
	NATIVO	CROSS-PLATFORM	HÍBRIDO
Performance	★★★★★	★★★★☆	★★★☆☆
Custo	Alto	Médio	Baixo
Manutenção	Separada	Única base	Média
Acesso a Recursos	Total	Quase total	Limitado

FERRAMENTAS POPULARES



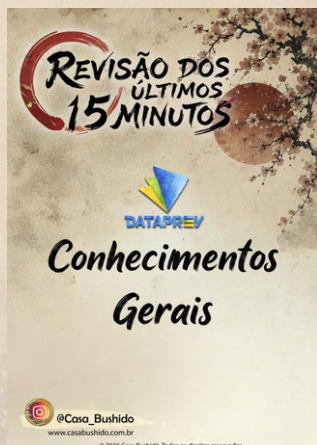
TENDÊNCIAS

- ✓ IA no Mobile (IA Generativa)
- ✓ Apps Offline First
- ✓ Realidade Aumentada (AR)
- ✓ PWA (Progressive Web Apps)



"BONS APPS NÃO SÃO SÓ CÓDIGO. SÃO EXPERIÊNCIAS!"

Jabá básico



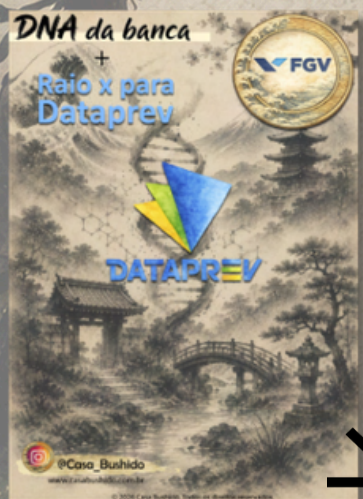
Clique aqui para
comprar

Veja também a
revisão dos 15
minutos para
conhecimentos
gerais do concurso
Dataprev!

Cupom por tempo limitado (10% off)

DATA PREV10

Preço normal: R\$ 22,90



Clique aqui para
comprar

Veja também o material
analisando a banca, falando
da Dataprev, dicas de
estudos para passar na frente
da concorrência, matérias
importantes e muito mais!

Cupom por tempo limitado (10% off)

DATA PREV10

Preço normal: R\$ 15,90

www.casabushido.com.br

Sugestões, críticas, ombro amigo, ajuda nos estudos. Podem contar comigo.
Entre em contato pelo canal do Instagram.



Casa Bushido

Com carinho
Vitor Scooby

