Apresentação do ambiente e conceitos básicos

Bem-vindos à primeira parte de nossa aula, dedicada à "Apresentação do Ambiente e Conceitos Básicos" na disciplina de Programação II. Iniciaremos nossa jornada explorando os fundamentos essenciais para o desenvolvimento de software, utilizando ferramentas como o Node.js, o Visual Studio Code (VS Code), o Postman, entre outras. Vamos nos aprofundar em cinco áreas-chave, começando pelo "Ambiente de Desenvolvimento". É crucial compreender o que é um ambiente de desenvolvimento e sua relevância para iniciar no campo da programação.

Prosseguimos com a introdução ao Node.js, dando uma visão sobre uma das linguagens de programação mais utilizadas atualmente. Este segmento é projetado para familiarizar você com as configurações e peculiaridades da linguagem. Discutiremos também o "Versionamento de Código", vital para a gestão de projetos de software, e ferramentas de teste de APIs, como o Postman, cruciais para validar a funcionalidade de aplicações. Por fim, exploraremos as IDEs, plugins e navegadores, ferramentas indispensáveis no desenvolvimento de software.

Avançaremos para a discussão sobre o "Versionamento de Código". Este tópico é vital para a gestão eficaz de projetos de software, permitindo a compreensão da importância de manter versões do seu trabalho organizadas e acessíveis.

Em seguida, abordaremos ferramentas de teste de APIs, como o Postman, essenciais para validar a funcionalidade de suas aplicações. Este

conhecimento é crucial para o desenvolvimento de software, garantindo que as aplicações se comportem como esperado em diferentes cenários.

Por fim, exploraremos o que são IDEs, plugins e navegadores, ferramentas indispensáveis no dia a dia do desenvolvimento de software. Entender como configurar e utilizar esses recursos otimiza o processo de desenvolvimento, permitindo um trabalho mais eficiente e produtivo.

Ambiente de desenvolvimento

Agora, focaremos no nosso primeiro tópico: o ambiente de desenvolvimento. Trata-se do conjunto de ferramentas e espaços de trabalho utilizados para desenvolver softwares. Esse ambiente engloba tudo o que é necessário para escrever, testar e depurar código. Inicialmente, este ambiente é local, configurado na própria máquina do desenvolvedor, mas é importante entender que existem outros tipos de ambientes, tais como ambientes de teste, treinamento, homologação e produção.

Cada um desses ambientes desempenha um papel crucial no ciclo de vida do desenvolvimento de software, desde a codificação inicial até a entrega do produto final. O ambiente local é o ponto de partida, onde o desenvolvimento começa na máquina do programador. O ambiente de desenvolvimento, por sua vez, permite testes mais abrangentes e colaboração entre desenvolvedores. Ambientes de teste e treinamento são configurados para garantir a qualidade e a funcionalidade do software antes de sua entrega. Por fim, os ambientes de homologação e produção servem como etapas finais, onde o software é validado pelo cliente e, posteriormente, disponibilizado ao usuário final.

Compreender a função e a importância de cada um desses ambientes é fundamental para o desenvolvimento eficaz e eficiente de aplicações.

Apresentação do Node.js

Prosseguindo em nossa jornada, nos debruçamos agora sobre o Node.js, um aspecto fundamental no desenvolvimento de aplicações modernas. Diferentemente de ambientes como o .NET para C# ou o JRE para Java, o Node.js atua como um ambiente de execução para a linguagem JavaScript fora do navegador. Enquanto outras plataformas são usadas para desenvolver aplicações para um contexto específico (como aplicações desktop ou sistemas empresariais), o Node.js se destaca por permitir o uso de JavaScript, tradicionalmente uma linguagem de front-end para web, em aplicações server-side.

Sua adoção massiva e a popularidade em constante crescimento se devem, em grande parte, à sua versatilidade e à baixa curva de aprendizado, especialmente para aqueles já familiarizados com JavaScript no desenvolvimento web. Node.js é um framework que transformou o JavaScript de uma linguagem primariamente usada para scripts de navegador em uma ferramenta full-stack, capaz de desenvolver tanto o cliente (front-end) quanto o servidor (back-end).

Diferentemente do JavaScript tradicional, que é executado no navegador e voltado para a criação de interações dinâmicas no lado do cliente, o JavaScript no Node.js é utilizado para construir a lógica do lado do servidor. Isso inclui a manipulação de requisições, processamento de dados, comunicação com bancos de dados e entrega de respostas ao cliente. Essa capacidade de executar JavaScript no lado do servidor revolucionou a maneira como desenvolvemos aplicações web, permitindo um desenvolvimento unificado e eficiente.

O Node.js facilita a integração entre front-end e back-end, promovendo uma maior coesão e sincronia no desenvolvimento de aplicações. O fluxo de trabalho envolve o envio de requisições pelo cliente e a geração de respostas pelo servidor, utilizando JavaScript em ambos os contextos, o que simplifica a arquitetura geral da aplicação e melhora a eficiência no desenvolvimento.

Para começar a usar o Node.js, é importante verificar sua presença e versão no sistema, o que pode ser feito através de um comando simples no terminal, 'node -v'. Caso não esteja instalado, o processo de adição do Node.js ao seu ambiente de desenvolvimento é direto e pode ser iniciado com o download da versão mais adequada do site oficial.

Este segmento inicial visa preparar todos para a programação com Node.js, destacando suas capacidades e potencial para construir aplicações robustas. Avançaremos na configuração do ambiente de desenvolvimento e no uso prático do Node.js, focando na aplicação do JavaScript em programação server-side.

Versionamento de código

Adentrando a terceira parte de nossa exploração, mergulhamos no essencial universo do versionamento de código, um conceito que se torna ainda mais relevante quando consideramos as diferentes aplicações do JavaScript no desenvolvimento web e mobile. O versionamento de código é uma pedra angular na prática do desenvolvimento de software, permitindo que equipes de desenvolvedores gerenciem eficientemente as alterações aplicadas ao longo do tempo em seus projetos. Essa gestão é crucial tanto para aplicações web, onde o JavaScript atua no navegador para criar interações dinâmicas, quanto para aplicações mobile, muitas vezes desenvolvidas com frameworks como React Native, que também utiliza JavaScript.

No desenvolvimento web, o JavaScript é executado dentro do navegador, facilitando a interação com o usuário e a manipulação dinâmica do conteúdo da página. Já no desenvolvimento mobile, o JavaScript pode ser empregado em frameworks que permitem a criação de aplicações nativas ou híbridas, operando em diferentes plataformas móveis. Em ambos os casos, o versionamento do código é vital para organizar, colaborar e evoluir

essas aplicações, garantindo que todas as alterações e melhorias estejam devidamente documentadas e controladas ao longo de seu ciclo de vida.

Desde a versão inicial (V0), passando por versões beta destinadas a testes de usuários, até versões alfa para uso interno, o versionamento de código permite que os desenvolvedores acompanhem e gerenciem as modificações no projeto. Isso é igualmente importante tanto para grandes sistemas quanto para projetos menores, abrangendo desde aplicações web complexas até apps mobile inovadores.

O versionamento ajuda a criar um ambiente seguro, onde as alterações podem ser testadas e validadas, promovendo uma colaboração eficiente e sem conflitos entre as equipes. Com sistemas de controle de versão como o local, o centralizado e o distribuído, as equipes podem escolher o método que melhor se adapta às suas necessidades de desenvolvimento e colaboração.

Especificamente, o versionamento de código distribuído destaca-se por sua flexibilidade e segurança, essenciais tanto no desenvolvimento web quanto no mobile, onde diversas alterações e atualizações são constantes. Essa abordagem facilita a integração e revisão das contribuições em um repositório comum, assegurando uma gestão eficiente das alterações em todas as etapas do desenvolvimento.

Entender e aplicar o versionamento de código é fundamental para a gestão de projetos de software, refletindo a necessidade de uma prática padrão na indústria, tanto para aplicações web quanto mobile. Vamos aprofundar nosso conhecimento no ambiente de desenvolvimento, com um foco especial em testes de aplicação, para solidificar e expandir nossa compreensão sobre como criar, manter e entregar software de qualidade em diferentes plataformas. Encorajo a todos a se familiarizarem com as ferramentas e práticas de versionamento, essenciais em sua jornada de desenvolvimento.

Postman e similares

Prosseguindo, adentramos no universo das ferramentas de teste, com ênfase no Postman e similares. Estas ferramentas são essenciais para qualquer desenvolvedor que trabalhe com APIs, proporcionando um meio eficaz de testar, documentar e colaborar no desenvolvimento de interfaces de programação de aplicações.

O Postman, uma das ferramentas mais populares e utilizadas na indústria, oferece uma interface intuitiva e recursos abrangentes para testes de APIs. Ao acessar o Postman, você se depara com uma tela composta por menus de ação, coleções de requisições de teste, estruturas de requisições de exemplo e áreas dedicadas à documentação e workspaces. Essa organização facilita o trabalho do desenvolvedor, permitindo uma visão clara do processo de teste e da documentação das APIs.

A funcionalidade principal do Postman é permitir que os desenvolvedores realizem requisições a aplicações (APIs ou não) e recebam respostas. Isso viabiliza uma série de testes, desde verificações básicas de funcionamento até testes automatizados complexos, garantindo a correta execução das funcionalidades da aplicação. Esses testes abrangem a validação do formato das requisições, a integridade das respostas e a documentação precisa de cada aspecto da API.

Além do Postman, existem outras ferramentas no mercado que oferecem funcionalidades similares, como Insomnia, Swagger, SoapUI e Apigee. Cada uma dessas ferramentas tem suas peculiaridades, vantagens e possíveis limitações. Algumas podem oferecer funcionalidades que o Postman não possui, enquanto outras podem se destacar pela facilidade de uso ou pela documentação disponível. É fundamental que os desenvolvedores explorem e testem essas ferramentas para identificar aquela que melhor atende às necessidades do projeto em questão.

Um ponto crítico a considerar na escolha de uma ferramenta de teste de API é a segurança dos dados manipulados. É imprescindível verificar onde e como os dados utilizados nos testes são armazenados, especialmente ao trabalhar com informações sensíveis. A escolha de uma ferramenta deve considerar a política de segurança da mesma, garantindo a proteção dos dados da empresa e a conformidade com normas e regulamentações vigentes.

IDE, plugins e navegadores

Concluindo, dedicaremos esta parte ao estudo de Ambientes de Desenvolvimento Integrados (IDEs), plugins e navegadores, elementos cruciais para o fechamento do nosso ambiente de desenvolvimento local. Com essas ferramentas, você estará plenamente equipado para embarcar nos projetos da disciplina.

As IDEs, sigla para Integrated Development Environments ou Ambientes de Desenvolvimento Integrados em português, são softwares completos que facilitam a vida do desenvolvedor. Essas plataformas suportam múltiplas linguagens de programação, como JavaScript, TypeScript, Java, C, entre outras, e oferecem uma gama de funcionalidades, incluindo edição de código, depuração, compilação e execução. Embora seja possível programar usando apenas um editor de texto simples, as IDEs proporcionam uma experiência muito mais rica e produtiva.

Para este curso, escolhemos o Visual Studio Code (VS Code) como nossa IDE principal devido à sua leveza, facilidade de uso e versatilidade. O VS Code, uma oferta da Microsoft, permite trabalhar confortavelmente em quase todos os sistemas operacionais e suporta uma ampla gama de plugins que estendem suas funcionalidades, adequando-se às necessidades específicas de cada projeto. Você pode encontrar um rápido passo a passo para sua instalação no fim deste material.

Falando em plugins, eles atuam como complementos que enriquecem a IDE, adaptando-a para linguagens específicas ou adicionando novas funcionalidades. Durante o curso, exploraremos alguns plugins essenciais, como o Code Runner, que simplifica a execução de programas; o SonarLint, que sugere melhorias e práticas de otimização de código; e o GitLens, que melhora a integração com o Git, oferecendo insights valiosos sobre o histórico de mudanças no código.

Além das IDEs e plugins, os navegadores são ferramentas indispensáveis no desenvolvimento web, servindo como a interface entre o usuário e as aplicações web. Durante o curso, nossa preferência será pelo Google Chrome, devido à sua ampla aceitação e compatibilidade com padrões web. Contudo, é importante familiarizar-se com outros navegadores como Microsoft Edge, Firefox e Safari, pois a escolha do navegador pode impactar tanto o desenvolvimento quanto a experiência do usuário final, considerando especialmente as diferenças na interpretação do JavaScript e na renderização de elementos front-end.

Ao final desta etapa de nossa jornada, você deve ter seu ambiente de desenvolvimento configurado com a IDE escolhida, os plugins recomendados e estar pronto para utilizar o navegador de sua preferência. Esta configuração é o ponto de partida para mergulharmos na prática e nos conceitos avançados de programação nos módulos seguintes, sempre buscando uma abordagem didática e engajadora. Para isso, teremos uma rápida sessão que auxiliará nessa configuração.

Configurando uma IDE, na prática

Para iniciar, vamos configurar o Node.js em seu sistema. Assumindo que você já tenha o Node.js instalado, crie uma pasta para o seu projeto e abra-a com o VS Code. Dentro do VS Code, abra o terminal integrado e execute o seguinte comando para iniciar um novo projeto Node.js:

```
npm init -y
```

Este comando cria um arquivo `package.json` com as configurações padrão do projeto. Agora, vamos criar um arquivo `index.js` como ponto de entrada do nosso aplicativo. Dentro deste arquivo, escreva um simples servidor HTTP com o Node.js:

```
const http = require('http');

const server = http.createServer((req, res) => {
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  res.end('Olá, Mundo!');
});

const PORT = 3000;

server.listen(PORT, () => {
  console.log(Servidor rodando na porta ${PORT});
});
```

Após salvar o arquivo, volte ao terminal integrado e execute o comando `node index.js` para iniciar o servidor. Seu servidor agora estará acessível no navegador pelo endereço `http://localhost:3000`, exibindo a mensagem "Olá, Mundo!".

Utilizando Plugins no VS Code para Melhorar a Produtividade

No VS Code, plugins como ESLint e Prettier podem ser instalados para melhorar a qualidade do código e a produtividade. ESLint ajuda a identificar erros de sintaxe e padrões de código não recomendados, enquanto o

Prettier formata automaticamente seu código conforme as regras de estilo definidas.

Para instalar esses plugins, acesse a aba de extensões no VS Code, procure por ESLint e Prettier e instale-os. Após a instalação, o VS Code aplicará automaticamente as regras de formatação e análise estática ao seu código, garantindo que ele esteja limpo e seguindo as melhores práticas.

Testando APIs com Postman

Supondo que o servidor criado anteriormente seja uma API simples, você pode usar o Postman para testar sua resposta. No Postman, crie uma nova requisição GET para o endereço `http://localhost:3000` e envie a requisição. Você receberá a resposta "Olá, Mundo!", confirmando que a API está funcionando como esperado.

Esse exemplo prático ilustra como configurar um ambiente de desenvolvimento básico para um projeto Node.js, utilizando o VS Code e plugins para aumentar a eficiência, além de demonstrar a utilização do Postman para testar APIs. Com essas habilidades, você estará bem preparado para iniciar o desenvolvimento de aplicações na disciplina, aplicando os conceitos básicos de ambiente de desenvolvimento, versionamento de código, e testes de API.

Encerramos, assim, a fase preparatória de nosso curso. Espero que essa aula tenha fornecido uma base sólida sobre as ferramentas e conceitos fundamentais de desenvolvimento. Nos próximos módulos, avançaremos para a prática, explorando a programação de maneira mais profunda e aplicada. Até lá, revigore seu entusiasmo e prepare-se para os desafios e aprendizados que virão. Nos vemos na próxima aula!

Instalação do VS Code

1. Acesse o site oficial: abra o navegador de sua preferência e visite o site oficial do Visual Studio Code.

2. Baixe o instalador: clique no botão de download correspondente ao seu sistema operacional (Windows, macOS ou Linux). O site geralmente detecta automaticamente o sistema operacional que você está usando e sugere a versão apropriada.

3. Instale o VS Code:

- Para Windows: após o download, abra o arquivo executável (.exe) e siga as instruções de instalação. Durante a instalação, você pode escolher as opções adicionais como adicionar um ícone na área de trabalho ou adicionar o VS Code ao PATH para acesso pelo terminal.

- Para macOS: abra o arquivo baixado (.dmg) e arraste o ícone do Visual Studio Code para a pasta de Aplicativos.

- Para Linux: dependendo da distribuição, você pode baixar um .deb para Debian/Ubuntu ou um .rpm para Fedora/SUSE. Depois de baixado, você pode instalar usando seu gerenciador de pacotes gráfico ou via terminal (por exemplo, usando dpkg -i code.deb para um arquivo .deb).

4. Execute o VS Code: após a instalação, abra o Visual Studio Code através do ícone na área de trabalho, menu Iniciar, ou terminal (executando o comando code).

5. Configuração inicial (opcional): ao abrir o VS Code pela primeira vez, você pode personalizar a configuração, como escolher o tema de cores, ajustar preferências de editor, e instalar extensões necessárias para a sua necessidade de desenvolvimento, como suporte a diferentes linguagens de programação, ferramentas de depuração etc.

Seguindo esses passos, você terá o Visual Studio Code instalado e pronto para uso no seu computador.

GitHub da Disciplina

Você pode acessar o repositório da disciplina no GitHub a partir do seguinte link: https://github.com/FaculdadeDescomplica/ProgramacaoII. Esse repositório tem como principal objetivo guardar os códigos das aulas práticas da disciplina para aprimorar suas habilidades em vários tópicos, incluindo a criação e consumo de APIs com controle de autenticação utilizando Node.js e utilizando boas práticas de programação e mercado.

Conteúdo Bônus

Para complementar nosso aprendizado sobre a aula e oferecer um guia prático inicial sobre o VS Code, recomendo o vídeo “Como Baixar e Configurar o Visual Studio Code Para Iniciantes” pelo canal Dev em Dobro no YouTube. Este conteúdo gratuito serve como um excelente recurso para os alunos que desejam visualizar os o primeiro contato com a IDE, proporcionando uma compreensão mais profunda visual desse início.

Referências Bibliográficas

Bibliografia Básica:

ELMASRI, R.; NAVATHE, S. B. **Sistemas de banco de dados.** 7.ed. Pearson: 2018.

MEDEIROS, L. F. de. **Banco de dados:** princípios e prática. Intersaberes: 2013.

VICCI, C. (Org.). **Banco de dados**. Pearson: 2014.

Bibliografia Complementar:

CARDOSO, L. da C. **Design de aplicativos**. Intersaberes: 2022.

JOÃO, B. do N. **Usabilidade e interface homem-máquina**. Pearson: 2017.

LEAL, G. C. L. **Linguagem, programação e banco de dados**: guia prático de aprendizagem. Intersaberes: 2015.

PUGA, S.; FRANÇA, E.; GOYA, M. **Banco de dados**: implementação em SQL, PL/SQL e Oracle 11g. Pearson: 2013.

SETZER, V. W.; SILVA, F. S. C. **Bancos de dados**. Blucher: 2005.

Ir para exercício