Introdução a Programação, Git e Github

SEMANA 1

APRESENTAÇÃO

01

ACORDOS E ALINHAMENTO DE EXPECTATIVAS

02

INTRODUÇÃO A PROGRAMAÇÃO

03

VERSIONAMENTO DE CÓDIGO, GIT E GITHUB

04

EXERCÍCIOS DA SEMANA

ACORDOS

- Este é um ambiente seguro, não tenha medo de perguntar.
- Pausas são importantes
- Feedbacks são sempre bem vindos!

VAMOS JUNTAS?!

INTRODUÇÃO A PROGRAMAÇÃO

hardware, software e programação

Os **componentes físicos** de um computador são chamados de **hardware** e possuem uma linguagem composta por **bits**, que são **zeros e uns**.

O **software** é o meio pelo qual a **linguagem de máquina** pode ser **compilada ou interpretada**, através de **códigos criados em uma linguagem intermediária**.

hardware, software e programação

A programação é exatamente quem **possibilita a existência dos softwares** e, por consequência, a **utilização mais prática dos hardwares**, já que permitem criar programas que **controlam o comportamento físico e lógico** de uma máquina.

hardware, software e programação

Esses programas, por sua vez, são compostos por **conjuntos de instruções** determinados que **descrevem tarefas a serem realizadas** pela máquina e atendem diversas finalidades, chamados de **algoritmos**.

você sabia?

A primeira pessoa programadora foi uma **mulher** chamada **Ada Lovelace**, que **escreveu um algoritmo para possibilitar a utilização da máquina analítica** de Charles Babbage, uma máquina robusta, de difícil comunicação, considerada a **precursora dos computadores eletrônicos** atuais.

MAS O QUE É UM ALGORITMO?

algoritmo

De acordo com o dicionário, é um **processo de cálculo** que, por meio de uma **sequência finita de operações,** aplicada a um **número finito de dados**, leva à **resolução de problemas**.

algoritmo

Podemos dividir um algoritmo em três fases fundamentais: **entrada, processamento e saída.**

Entrada recebe as **informações necessárias para iniciar** nosso algoritmo; Processamento são os **passos necessários para atingir nosso objetivo**; Saída é o **resultado esperado** da fase de processamento.

VAMOS PRATICAR?

exercícios

1. Faça um algoritmo que mostre o passo a passo para trocar uma de lâmpada queimada.
2. Faça um algoritmo que mostre o passo a passo para passear com seu animal de estimação.
3. Faça um algoritmo que mostre o passo a passo para acessar um computador.
4. Faça um algoritmo que mostre o passo a passo para lavar um copo
5. Faça um algoritmo que mostre o passo a passo para postar uma foto em um rede social

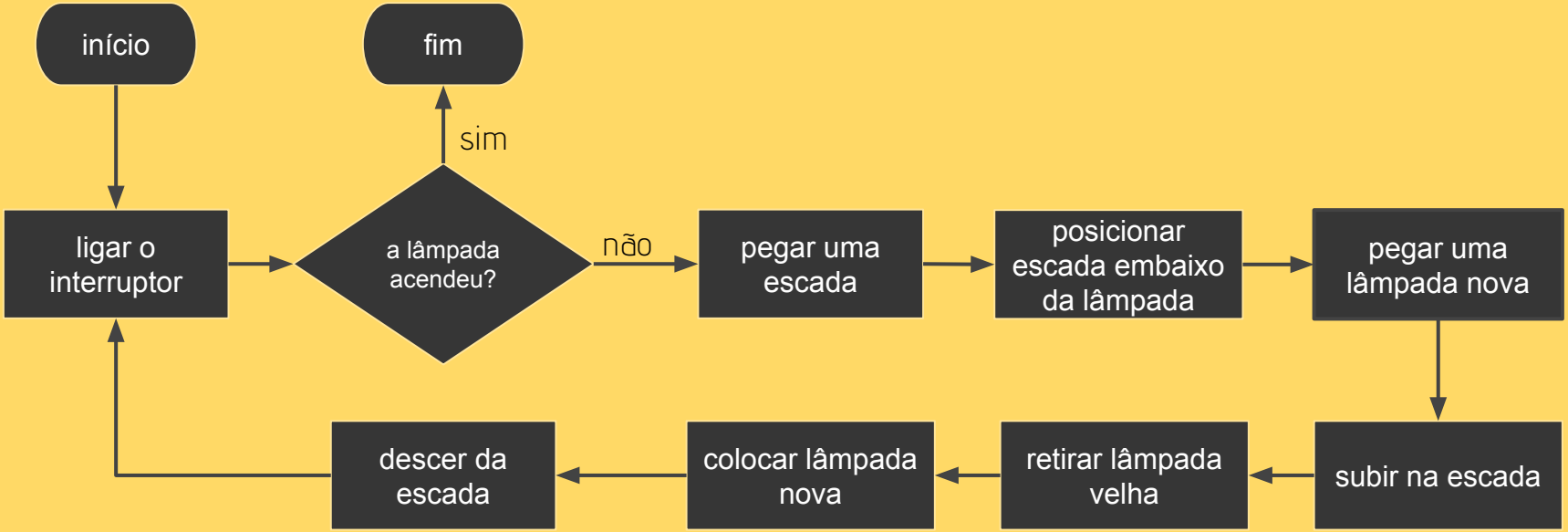
outras formas de se escrever um algoritmo

Há outras formas de se escrever um algoritmo além do formato de **narrativa**. Também é possível criar um **fluxograma** ou um **pseudocódigo**.

fluxograma

Um fluxograma é a **representação gráfica de um procedimento, problema ou sistema**, cujas etapas ou módulos são ilustrados de forma encadeada por meio de **símbolos geométricos** interconectados.

fluxograma - exemplo



pseudocódigo (ou portugol)

Pseudocódigo é uma **forma genérica de escrever um algoritmo**, utilizando uma **linguagem simples** sem necessidade de conhecer a sintaxe de uma linguagem de programação.
Exemplo: decidir se a pessoa deve votar em uma eleição ou não

```
recebe um inteiro idade
idade é igual a 29
se idade é maior ou igual a 18 e menor que 70:
        seu voto é obrigatório
se não, seu voto é facultativo
```

LINGUAGEM DE PROGRAMAÇÃO

linguagem de programação

Os algoritmos que escrevemos até aqui ainda não podem ser interpretados pela máquina, já que **não estamos falando o mesmo "idioma".**

Pra que isso funcione, precisamos de uma **linguagem de programação** que faça a ponte entre a **nossa comunicação e a do computador**.

front-end

Front-end é a **parte visual de uma aplicação**, onde é possível que a usuária **interaja com o sistema por meio de uma interface gráfica**.

As **tecnologias base** para desenvolvimento front-end na web são **JavaScript, HTML e CSS**, mas também existem diversas **bibliotecas e frameworks** em constante evolução.

back-end

Back-end, como o próprio nome sugere, vem da ideia **do que tem por trás de uma aplicação**. É a parte responsável pela **implementação das regras de negócio** e que **não interage** com a usuária diretamente. Também pode ser responsável por fazer a **ponte entre os dados que vem do navegador e o banco de dados.**

Tudo isso pode ser feito com uma série de linguagens como Python, Ruby, Java, C#, **JavaScript (Node.js)** entre outras.

banco de dados

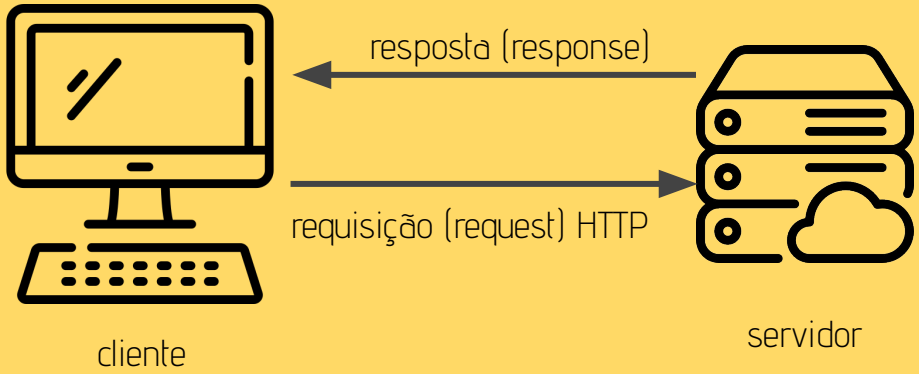
O banco de dados é a **organização e armazenagem de informações sobre um domínio específico**. De forma mais simples, é o **agrupamento de dados que tratam do mesmo assunto**, e que precisam ser armazenados para **segurança** ou **conferência futura**.

arquitetura cliente servidor

Uma aplicação web é composta por dois atores principais: **cliente e servidor**.

O cliente normalmente é um **navegador** como o Internet Explorer ou Firefox. O servidor é uma **aplicação**, na forma de um serviço, normalmente hospedado remotamente.

arquitetura cliente servidor



você sabia?

Hedy Lamarr, **inventora e atriz de Hollywood,** criou um **sistema de comunicações** para as Forças Armadas dos EUA que serviu como **base para a invenção do Wi-Fi** e também da atual **telefonia celular**.

CONTROLE DE VERSÃO, GIT E GITHUB

controle de versão

O controle de versão consiste basicamente em um **sistema que permite registrar alterações** feitas no desenvolvimento de um software.

É a partir dele que toda a equipe envolvida no projeto têm acesso ao **histórico das versões anteriores** do software, podendo **recuperar uma versão específica** ou **compreender quais mudanças foram feitas** por outras pessoas.

git

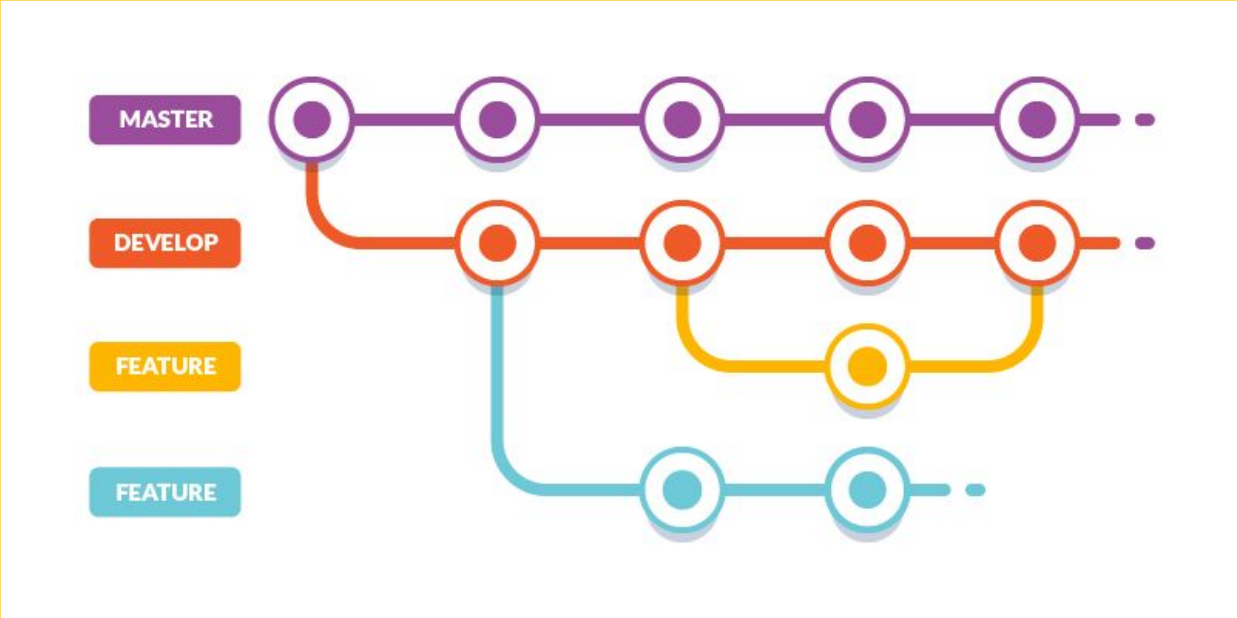
Git é um **sistema de controle de versões**, usado principalmente no desenvolvimento de software.

O Git é um **software livre** e foi inicialmente projetado e desenvolvido por **Linus Torvalds** para o desenvolvimento do kernel Linux.

git

Desde seu nascimento em 2005, Git evoluiu e amadureceu para ser fácil de usar. Ele é **incrivelmente rápido**, é muito **eficiente com projetos grandes**, e ele tem um incrível sistema de ramos (*branchs*) para desenvolvimento não linear.

git



ANTES DE CONTINUAR...

terminal

Basicamente, terminal é aquela **famosa tela preta** na qual você **digita comandos para dar instruções para um computador**. Ou seja, ele serve para você **executar tarefas no computador sem utilizar a interface gráfica**, com pastinhas e ícones, ou o bom e velho mouse. **Todos os comandos são executados através de digitação de texto puro.**

terminal - comandos básicos

ls (macOS/Linux) dir (Windows)	lista todos os arquivos presentes no diretório atual (macOS/Linux)
mkdir nome-da-pasta	cria uma nova pasta
cd nome-da-pasta	navega para a pasta especificada (exemplo: *cd documentos*)
cd ..	sobe um nível de pasta
touch nome-do-arquivo dir > nome-do-arquivo	cria um novo arquivo
clear	limpa todas as informações do terminal

CONTINUANDO...

git - configuração inicial

1. Instale o Git na sua máquina
2. Configure suas informações (nome e email)
    a. git config --global user.name "Seu Nome"
    b. git config --global user.email "Seu Email"
3. Verifique suas informações
    a. git config --list **OU**
    b. git config user.name **OU**
    c. git config user.email

git - comandos básicos

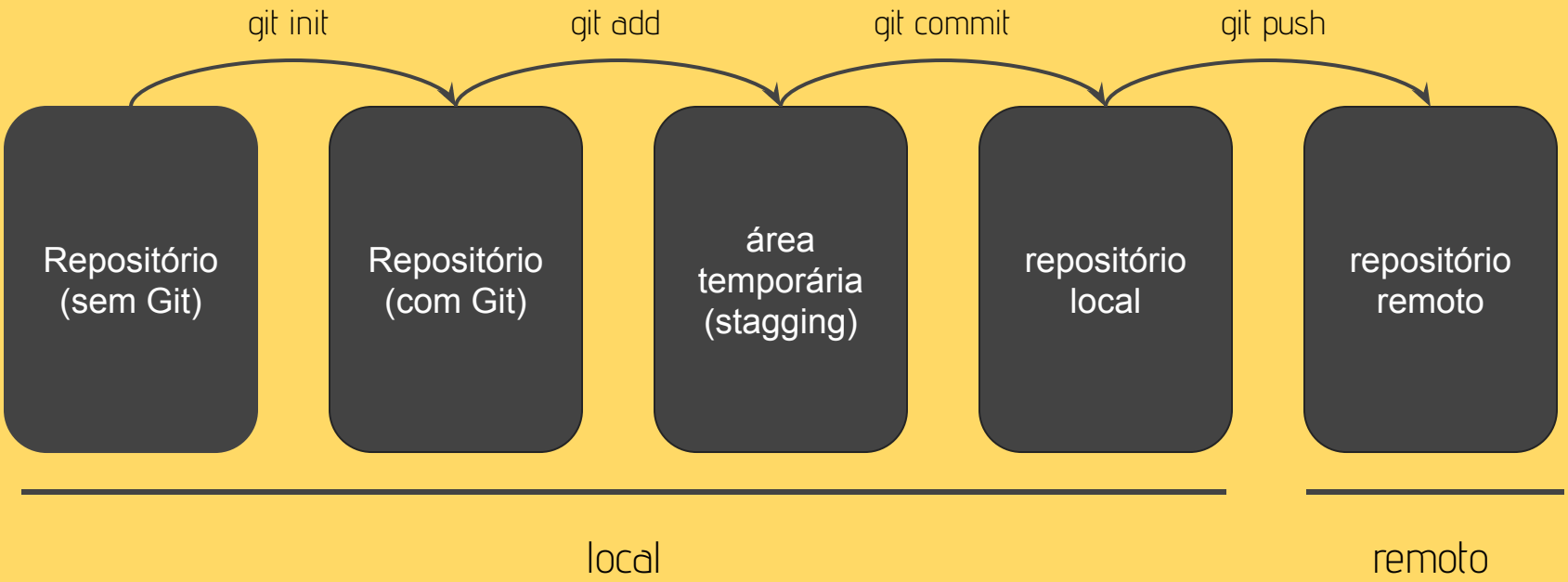
git init	inicializa o git no repositório local
git add	adiciona um arquivo modificado ao *stagging* (área temporária)
git status	mostra os status dos arquivos modificados
git commit -m "<*mensagem*>"	cria um commit
git pull	puxa as atualizações mais recente (remoto -> local)
git push	envia as atualizações mais recentes (local -> remoto)
git remote add origin <*caminho*>	adiciona o seu repositório local ao remoto
git checkout -- <*nome-arquivo*>	descarta as alterações locais do arquivo informado

VAMOS PRATICAR?

exercício

1. Começando com Git, **no terminal**
    a. Crie uma pasta
    b. Navegue até a pasta e inicialize o git
    c. Crie um arquivo qualquer e verifique seu status
    d. Adicione o arquivo ao stage do Git
    e. Faça um commit
    f. Faça um push

git


git init
git add
git commit
git push
Repositório (sem Git)
Repositório (com Git)
área temporária (stagging)
repositório local
repositório remoto
local
remoto

github

GitHub é uma **plataforma de hospedagem de código-fonte com controle de versão usando o Git**.

Ele permite que **qualquer pessoa cadastrada** na plataforma **contribua em projetos** privados e/ou de código-fonte aberto (Open Source) de **qualquer lugar do mundo**.

dicas importantes

1. As mensagens do commit devem ser claras e sucintas, descrevendo bem a alteração
2. Um push só funciona se houver um commit feito!
3. Não se desespere, Git pode parecer complicado no começo mas depois você faz de olhos fechados (ou quase isso haha)

exercícios

1. **Apresentação**
    a. Crie um repositório localmente e inicialize o git
    b. Adicione um arquivo **markdown** chamado README com seu nome e prato favorito e faça um *commit*
    c. Adicione uma curiosidade sobre você e faça outro *commit*
    d. Publique o repositório no seu GitHub

exercícios

2. **Algoritmos**
    a. Faça um *fork* do **repositório**.
    b. Clone o repositório para a sua máquina.
    c. Crie uma nova *branch* com seu nome (exemplo: yasminn-vaz).
    d. Faça *commits* com a resolução dos exercícios.
    e. Atualize seu repositório remoto.

3. **DESAFIO EXTRA**:
    a. Abra um *Pull Request* para o repositório original

OBRIGADA!

FERRAMENTAS

- Bloco de notas compartilhado: http://dontpad.com/
- Fluxograma: https://app.diagrams.net/
- GitHub: https://github.com/

REFERÊNCIAS

- Artigos:
  - O que é programação e qual sua importância para o futuro digital
  - Mulheres Históricas
  - O que é front-end e back-end?
  - Banco de Dados
  - Como criar um pull-request no GitHub
  - Git - Livro
  - Entendendo sobre branch e pull request
  - O que é o terminal

REFERÊNCIAS

- Artigos:
  - O que é controle de versão de software e como usar no seu projeto
  - Windows: introdução ao prompt

REFERÊNCIAS

- Vídeos:
  - O que é um algoritmo?
  - O que é uma linguagem de programação?
  - Como funciona a internet
  - Github
  - Git