O que é Fluxograma?

Um fluxograma é uma representação gráfica de um processo, sistema ou algoritmo, usando símbolos específicos para ilustrar os passos sequenciais e a lógica envolvida.

Ele é utilizado para mapear visualmente o fluxo de atividades ou operações, facilitando a compreensão e análise de processos complexos.

Benefícios de um Fluxograma

Um fluxograma vai além de uma simples ferramenta de organização de processos, ele também auxilia na tomada de decisão, na identificação de falhas, na criação de estratégias e planejamentos e na otimização de tarefas e atividades internas.

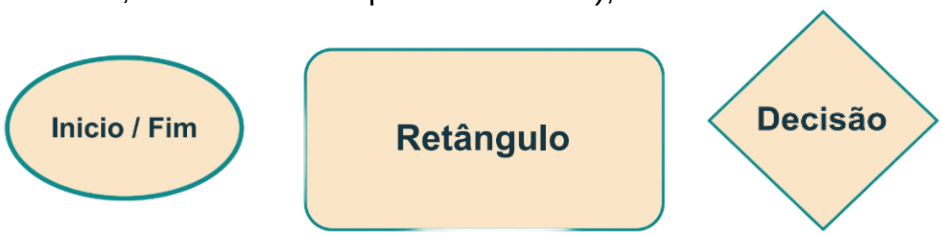
Os benefícios de um fluxograma incluem:

- Representação visual: o fluxograma fornece uma representação visual de um conceito e pode ajudar a torná-lo mais claro.
- Organização: os fluxogramas ajudam a manter um projeto organizado.
- Flexibilidade e controle: um fluxograma é relativamente fácil de se fazer.
- Clareza visual: um dos maiores benefícios de um fluxograma é a capacidade da ferramenta de apresentar vários progressos e as suas sequências num só documento.
- Comunicação instantânea: as equipes podem usar fluxogramas para substituir reuniões.

Como fazer um fluxograma?

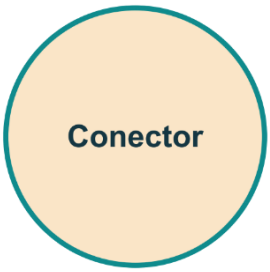
Os principais componentes de um fluxograma incluem:

- Formas e Símbolos: Cada tipo de atividade é representado por um símbolo específico (por exemplo, retângulos para atividades/processos, losangos para decisões, e círculos/ovais para início e fim);



- Setas: Indicando a direção do fluxo do processo;

- Conectores: Linhas que conectam os símbolos, mostrando o fluxo de informações ou tarefas.



Fluxogramas são amplamente utilizados em diversas áreas, como gestão de projetos, engenharia, desenvolvimento de software, e processos de negócios, para documentar, analisar, melhorar e comunicar processos.

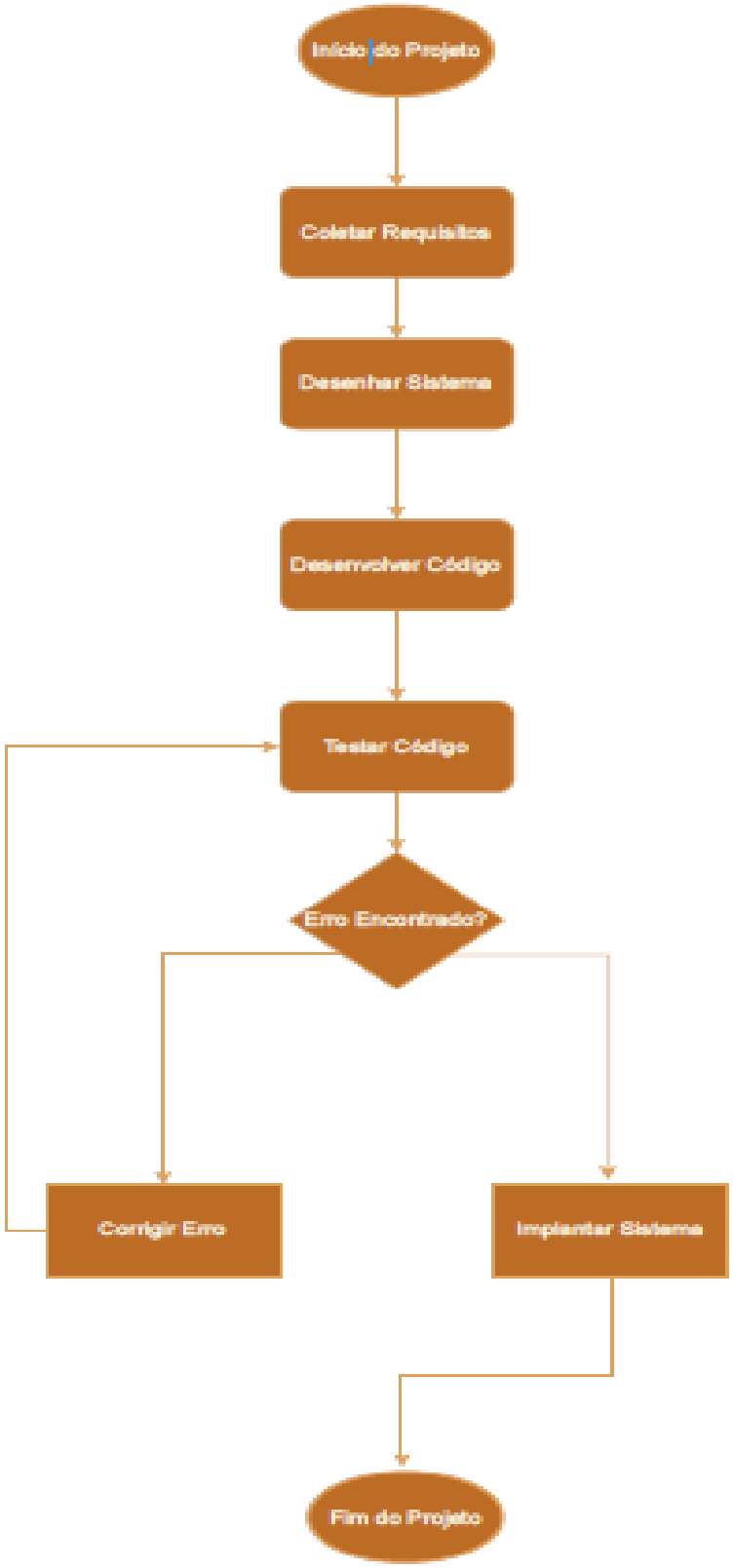
Existem alguns exemplos comuns de fluxogramas em diferentes contextos:

- Fluxograma de Processo de Negócios;
- Fluxograma de Suporte ao Cliente;
- Fluxograma de Fabricação.

Nós também temos o fluxograma voltado para área de Desenvolvimento de Software. Abaixo mostramos um exemplo:

- Início: Oval com "Início do Projeto";
- Análise de Requisitos: Retângulo com "Coletar Requisitos";
- Desenho de Sistema: Retângulo com "Desenhar Sistema";
- Codificação: Retângulo com "Desenvolver Código";
- Testes: Retângulo com "Testar Código";
- Decisão: Losango com "Erro Encontrado?";
- Correção de Erro: Retângulo com "Corrigir Erro" (conectando de volta aos testes);
- Implantação: Retângulo com "Implantar Sistema";
- Fim: Oval com "Fim do Projeto".

Exemplo do Fluxograma de Desenvolvimento de Software:



O que é documentação de software?

É uma etapa do desenvolvimento do produto que consiste em registrar em texto e de forma precisa o que há de essencial a saber sobre um sistema/software.

Ela é um conjunto de documentos que explicam e detalham diversos aspectos de um software. Esta documentação é essencial para garantir que os usuários, desenvolvedores e outras partes interessadas entendam como usar, manter e desenvolver o software.

Qual a importância da documentação de software?

A documentação de software é importante porque:

- Melhora a qualidade e a confiabilidade do software;
- Reduz os erros e as falhas no funcionamento do software;
- Aumenta a produtividade e a eficiência do desenvolvimento e da entrega do software;
- Aumenta a satisfação e a fidelização dos clientes e dos usuários;
- Permite que todos tenham acesso a informações mais profundas sobre o código e toda a arquitetura do software.

Quando se tem a documentação técnica correta, as pessoas de produto podem tomar decisões de forma muito mais rápida. Além disso, ela facilita as tarefas de suporte e manutenção dos softwares.

Como fazer uma documentação de software?

Documente o que foi testado: Pode ser que um cliente que use nosso sistema há um bom tempo faça uso de um cenário que não temos imaginado – e isso não fica nada bem para a nossa imagem.

Documente ou não aconteceu: Você já teve aquela sensação de que uma feature foi lançada e logo depois esquecida? Se isso acontece e não há nada citado na documentação, é como se aquilo nunca tivesse existido.

A documentação de software geralmente inclui:

1. Documentação do Usuário:

- Manuais de Usuário: Guias detalhados que explicam como usar o software, com instruções passo a passo e capturas de tela;
- Tutoriais e Guias Rápidos: Instruções simplificadas para realizar tarefas comuns;
- FAQs e Solução de Problemas: Respostas para perguntas frequentes e dicas para resolver problemas comuns.

2. Documentação Técnica:

- Documentação do Desenvolvedor: Informações sobre a arquitetura do software, padrões de codificação, estruturas de dados, algoritmos, e exemplos de código;
- Documentação da API: Descrição detalhada das interfaces de programação de aplicativos, incluindo endpoints, parâmetros, exemplos de solicitações e respostas;
- Diagrama de Arquitetura: Representações visuais da estrutura do sistema, mostrando componentes e suas interações.

3. Documentação de Processo:

- Planos de Projeto: Detalhes sobre o escopo do projeto, cronogramas, recursos e tarefas;
- Especificações de Requisitos: Descrições das funcionalidades e características que o software deve ter;
- Casos de Uso: Cenários específicos que descrevem como o software será usado para alcançar determinados objetivos.

4. Documentação de Manutenção:

- Guia de Instalação: Instruções para instalar e configurar o software em diferentes ambientes;
- Guia de Atualização: Procedimentos para atualizar o software, incluindo migrações de dados e compatibilidade de versões;

- Registros de Alterações (Changelog): Lista de alterações feitas em cada versão do software, incluindo correções de bugs e novas funcionalidades.

5. Documentação de Testes:

- Planos de Teste: Descrição das estratégias de teste, escopo, recursos e cronogramas;
- Casos de Teste: Detalhes de cada teste específico, incluindo pré-condições, passos e resultados esperados;
- Relatórios de Teste: Resultados dos testes realizados, incluindo quaisquer defeitos encontrados e seu status.

A documentação de software é fundamental para a comunicação eficaz entre equipes de desenvolvimento, para a formação de novos membros da equipe, e para garantir que o software possa ser mantido e melhorado ao longo do tempo.

É importante que o documento seja claro, preciso e sem recorrer a tecnicismos que prejudiquem esse objetivo. Todos devem conseguir compreender com facilidade o que está na documentação.

Documentação Interna vs Externa

A documentação inteira é direcionada aos membros da equipe de desenvolvimento e manutenção do software. Ela tem como objetivo fornecer informações detalhadas sobre a estrutura interna do software, como arquitetura, design, algoritmos, interfaces de programação, entre outros. A documentação interna é essencial para que os desenvolvedores possam entender a trabalhar no código-fonte do software, facilitando a colaboração e a manutenção sistema.

Por outro lado, a documentação externa é voltada para os usuários finais do software. Ela tem como objetivo fornecer informações sobre como utilizar o software de forma correta e eficiente, incluindo manuais do usuário, guias de instalação, tutorias, entre outros. A documentação externa deve ser clara, concisa e de fácil compreensão, visando ajudar os usuários a aproveitar ao máximo as funcionalidades do software.

A documentação interna e externa são complementares e desempenham papéis diferentes no ciclo de vida do software.

Documentação de código (comentários, docstrings)

A documentação de código é um aspecto essencial do desenvolvimento de software. Ela envolve o uso de comentários e docstrings para explicar o funcionamento do código, tornando-o mais compreensível e facilitando a manutenção futura.

Os comentários são trechos de texto inseridos no código para fornecer informações adicionais sobre o que está acontecendo em determinada parte do programa. Eles são destinados aos desenvolvedores que precisam entender e modificar o código. Os comentários podem incluir explicações sobre a lógica do algoritmo, decisões de design, problemas conhecidos e outras informações relevantes. É importante escrever comentários claros e concisos, evitando redundâncias e informações óbvias.

As docstrings, por outro lado, são uma forma de documentação incorporada diretamente no código-fonte, geralmente associada a funções, classes e módulos. Elas são como strings de texto logo após a declaração da função ou classe e fornecem informações sobre o propósito, os parâmetros, os valores de retorno e o comportamento esperado. As docstrings podem ser extraídas automaticamente para gerar documentação externa, como páginas de ajuda ou manuais do usuário.

A documentação de código, seja por meio de comentários ou docstrings, é importante por vários motivos. Ela ajuda tomar o código mais legível, importante por vários motivos. Ela ajuda a tomar o código mais legível, facilitando a compreensão do seu propósito e funcionamento. Além disso, a documentação adequada permite que outros desenvolvedores trabalhem de forma mais eficiente no código existente, reduzindo a curva de aprendizado e facilitando a compreensão do seu propósito e funcionamento. Além disso, a documentação adequada permite que outros desenvolvedores trabalhem de forma mais eficiente no código existente, reduzindo a curva de aprendizado e facilitando a colaboração. Também é uma prática recomendada para garantir a manutenibilidade e a sustentabilidade do software ao longo do tempo.

Portanto, é essencial investir tempo e esforço na documentação de código, seguindo as melhores práticas e padrões estabelecidos pela comunidade de desenvolvimento. Isso ajudará a melhorar a qualidade do software e a facilitar o trabalho de todos os envolvidos no projeto.

Documentação de Usuário

Manuais: Os manuais são documentos detalhados que fornecem instruções passo a passo sobre como usar o software. Eles devem abordar todas as funcionalidades principais do sistema e fornece exemplos e ilustrações claras. Além disso, os manuais devem incluir informações sobre os requisitos do sistema, configuração inicial, solução de problemas comuns e dicas úteis. É importante que os manuais sejam escritos em linguagem clara e acessível, evitando jargões técnicos desnecessários.

Guias de uso: Os guias de uso são documentos mais concisos que se concentram em tarefas específicas ou funcionalidades do software. Eles são mais direcionados e podem ser usados como referência rápida para os usuários. Os guias de uso devem fornecer uma visão geral da funcionalidade, passos claros sobre como realizar a tarefa desejada e possíveis opções ou configurações relevantes. Eles também podem incluir imagens ou capturas de tela para ilustrar os passos.

Interface do usuário: Além dos documentos escritos, é importante considerar a interface do usuário do software. Certifique-se de que a interface seja intuitiva e fácil de usar, com rótulos claros, ícones significativos e fluxos de trabalho lógicos. Isso ajudará os usuários a navegar e usar o software sem depender exclusivamente da documentação.

Atualização contínua: A documentação do usuário final deve ser atualizada regularmente para refletir quaisquer alterações ou atualizações do software. À medida que novas versões são lançadas ou novas funcionalidades são adicionadas, a documentação deve ser revisada e atualizada para garantir que esteja atualizada e precisa.

Conclusões finais

Os fluxogramas complementam a documentação de software ao fornecer uma representação visual dos processos e fluxos de trabalho. Eles ajudam a ilustrar como diferentes componentes interagem e se interconectam, facilitando a compreensão para desenvolvedores, gerentes de projeto e outras partes interessadas. Já a documentação de software complementa fluxogramas ao fornecer detalhes textuais e explicativos que não podem ser facilmente capturados apenas por diagramas visuais.

Bibliografia

https://blog.vindi.com.br/documentacao-de-software/

https://asana.com/pt/resources/what-is-a-flowchart

https://miro.com/pt/fluxograma/o-que-e-fluxograma/