Agilidade Máxima

Dominando Metodologias Ágeis na Programação

ANDRÉA SOUZA

01

Scrum

Scrum
Organizando o Trabalho com Sprints

- O Scrum é uma das metodologias ágeis mais populares, que organiza o trabalho em ciclos chamados sprints. Cada sprint é uma janela de tempo onde um conjunto de tarefas é concluído. A ideia é sempre entregar algo funcional ao final de cada ciclo.
- **Exemplo de código**: Imagine que você tem uma tarefa de implementar uma funcionalidade de login. Em vez de esperar meses para entregar tudo de uma vez, você divide em pequenas entregas durante vários sprints.

- # Sprint 1: Criação da tela de login
- def mostrar_tela_login():
- print("Tela de Login")
- # Sprint 2: Validação de usuário e senha
- def validar_usuario(usuario, senha):
- if usuario == "admin" and senha == "senha123":
- return True
- return False

Scrum
Organizando o Trabalho com Sprints

- Durante o Sprint 1, você foca em entregar a interface. No Sprint 2, a funcionalidade de validação é construída. Cada entrega é pequena, mas funcional.

02

Kanban

Kanban
Fluxo Visual do Trabalho

- O Kanban é uma prática ágil que organiza as tarefas em um fluxo visual. Imagine um quadro onde as tarefas são movidas de uma coluna para outra, conforme o progresso. Com o Kanban, você consegue visualizar facilmente o que está "por fazer", "em andamento" e "concluído".
- **Exemplo de código**: Vamos criar uma simulação de um quadro Kanban simples para um projeto de software.
- # Kanban Simples
- tarefas = {
- "Por Fazer": ["Implementar login", "Criar banco de dados"],
- "Em Andamento": ["Desenvolver página de perfil"],
- "Concluído": ["Tela de cadastro"]
- }
- # Movendo uma tarefa
- def mover_tarefa(tarefa, de, para):
- if tarefa in tarefas[de]: tarefas[de].remove(tarefa) tarefas[para].append(tarefa)
- mover_tarefa("Implementar login", "Por Fazer", "Em Andamento")
- print(tarefas)

Kanban
Fluxo Visual do Trabalho

- O código acima gerencia as tarefas em um simples quadro Kanban, permitindo que você mova as tarefas conforme o progresso.

03

XP (Extreme Programming)

XP (ExtremeProgramming)
Testes Contínuos e Refatoração

- O XP é uma prática que foca em escrever código de qualidade desde o início. Isso inclui práticas como desenvolvimento orientado a testes (TDD), refatoração constante e programação em par.
- **Exemplo de código**: Vamos aplicar TDD para escrever uma função simples que calcula a soma de dois números.
- # Teste: deve somar dois números corretamente
- def test_soma():
- assert soma(2, 3) == 5
- assert soma(-1, 1) == 0
- assert soma(0, 0) == 0
- def soma(a, b):
- return a + b
- # Executando os testes
- test_soma()

XP (ExtremeProgramming)
Testes Contínuos e Refatoração

- No exemplo acima, primeiro escrevemos o teste para a função soma e, em seguida, implementamos a função para passar no teste. Isso assegura que o código está funcionando conforme esperado.

04

Kanban e Scrum Juntos

Kanban e Scrum Juntos

Combinando Abordagens

- Embora Scrum e Kanban sejam práticas distintas, muitas equipes combinam ambos para gerenciar o fluxo de trabalho. O Scrum organiza o trabalho em sprints, enquanto o Kanban ajuda a visualizar o progresso.
- **Exemplo de código**: Aqui, vamos criar uma simples simulação de um quadro Kanban dentro de um sprint.

```
# Simulação de Kanban + Scrum
sprint = {
    "Sprint 1": {
        "Por Fazer": ["Implementar login", "Criar banco de dados"],
        "Em Andamento": ["Desenvolver API de usuários"],
        "Concluído": []
    }
}
```

Kanban e Scrum Juntos
Combinando Abordagens

(continuação do código)

- # Movimento de tarefas
  def mover_tarefa_sprint(tarefa, sprint_nome, de, para):
  if tarefa in sprint[sprint_nome][de]:
  sprint[sprint_nome][de].remove(tarefa)
  sprint[sprint_nome][para].append(tarefa)

  mover_tarefa_sprint("Implementar login", "Sprint 1", "Por Fazer", "Em Andamento")
  print(sprint)

- Esse exemplo combina a organização de sprints com o fluxo visual do Kanban, gerenciando as tarefas de forma eficiente.

05

Lean

Lean
Eliminar Desperdícios e Maximizar Valor

- A metodologia Lean, que busca eliminar desperdícios, pode ser aplicada ao desenvolvimento ágil ao garantir que todo o esforço esteja focado em gerar valor para o cliente.
- **Exemplo de código**: Suponha que você tenha uma função de cadastro de usuário. Ao aplicar Lean, você removeria qualquer código ou etapa desnecessária.
- # Função de cadastro otimizada
- def cadastrar_usuario(usuario, senha):
- if not usuario or not senha:
- raise ValueError("Usuário e senha são obrigatórios")
- print(f"Usuário {usuario} cadastrado com sucesso!")
- # Teste
- cadastrar_usuario("admin", "senha123")
-

Lean
Eliminar Desperdícios e Maximizar Valor

- A função foi simplificada para evitar validações desnecessárias, mantendo apenas o que é essencial para o processo de cadastro.

Agradecimento

Quero agradecer sinceramente por você ter acompanhado este ebook. Sua dedicação em aprender sobre metodologias ágeis demonstra seu compromisso em melhorar como profissional e aprimorar seus projetos. A jornada de aprendizado é contínua, e espero que as práticas ágeis apresentadas aqui possam ser úteis para otimizar seu trabalho e trazer mais resultados para você e sua equipe.

Fico feliz em ter contribuído para seu desenvolvimento e desejo muito sucesso em sua jornada de implementação da agilidade no seu código. Continue sempre em busca de evolução!

Muito obrigado e até a próxima!