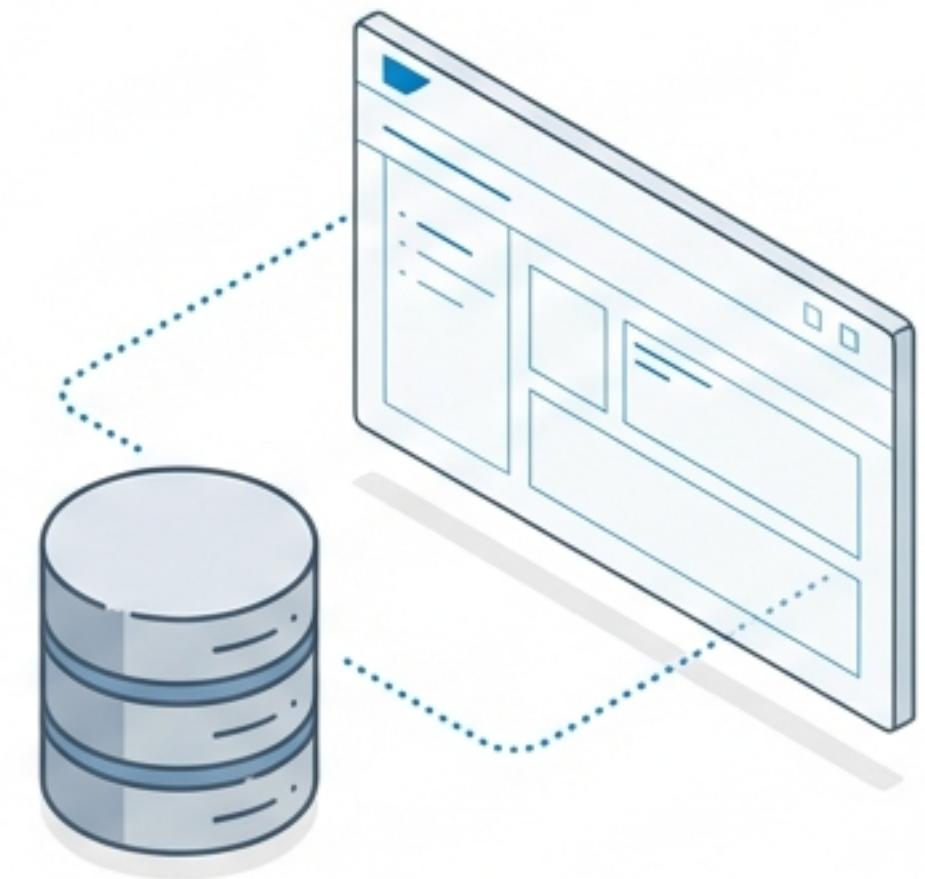
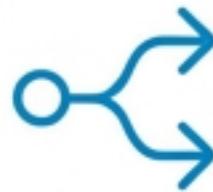


# **Definindo Meta Objetos (Metadata Extensions)**

Arquitetura Limpa e Separação de  
Preocupações no SAP RAP



# Objetivos de Aprendizagem



**Arquitetura:** Compreender e aplicar o princípio de 'Separação de Preocupações' (Lógica de Dados vs. Lógica de Apresentação).



**Configuração:** Habilitar CDS Views para enriquecimento externo com `@Metadata.allowExtensions: true`.



**Implementação:** Criar arquivos MDE (.ddlx) para centralizar anotações de UI `(@UI)`.



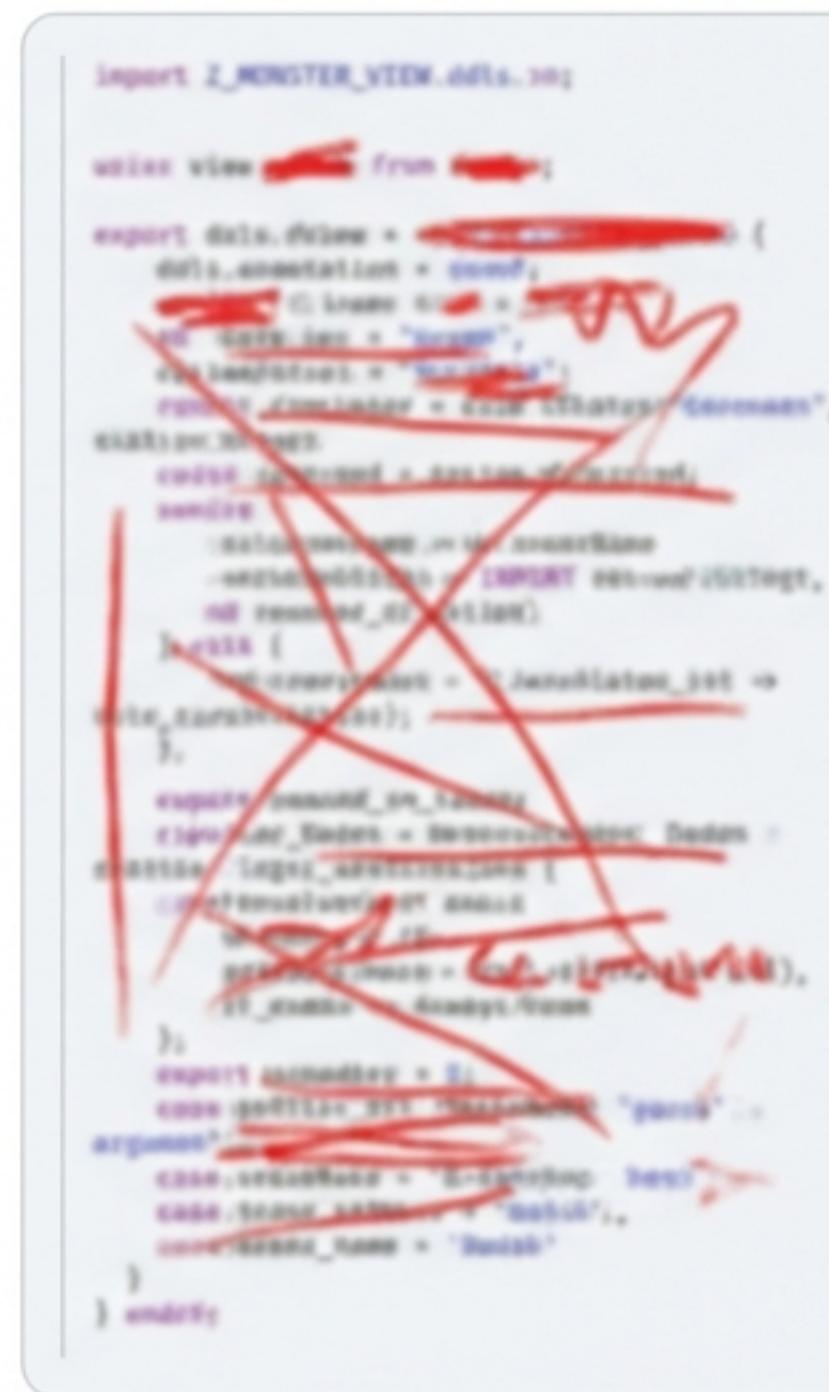
**Interface:** Dominar propriedades como `lineItem`, `selectionField`, `importance` e `criticality`.



**Hierarquia:** Entender o conceito de Layering (#CORE, #PARTNER, #CUSTOMER).

# O Perigo do Acoplamento: A “CDS View Monstro”

## Z\_MONSTER\_VIEW.dds



## Os 3 Riscos Principais



**Poluição Visual:** O Backend developer não consegue enxergar a lógica de negócio ou o modelo de dados.



**Rigidez:** Anotações 'chumbadas' (hardcoded) impedem o reuso da View para diferentes apps (Desktop vs. Mobile).



**Conflitos de Versão:** Desenvolvedores de UI e de Dados editam o mesmo arquivo simultaneamente.

Resultado: Arquivos com milhares de linhas e manutenção dolorosa.

# A Solução: Metadata Extensions (MDE)

Backend / Estrutura



**CDS View (.ddls)**

A 'Verdade do Dado'.

Contém: SQL, Associações,  
Lógica de Cálculo.



Frontend / Estilo



**Metadata Extension (.ddlx)**

O 'Estilo do Dado'.

Contém: Anotações @UI,  
Layout, Labels.

Assim como o CSS limpa o HTML, o MDE limpa a CDS View.

# Habilitando Extensões: O Contrato de Flexibilidade

Views são fechadas por padrão. É preciso autorizar o enriquecimento.

```
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Projection View for Travel'
/* A LINHA MÁGICA: */
@Metadata.allowExtensions: true
define view entity Z_C_TRAVEL as select from ...
```



**Atenção:** Se o arquivo MDE estiver perfeito, mas o Fiori exibir a tela 'crua' (sem formatação), 90% das vezes o erro é a ausência de @Metadata.allowExtensions: true. O framework falha silenciosamente.

# Mão na Massa: Criando o Arquivo MDE

1. No ADT (Eclipse), clique com o botão direito no pacote.
2. Selecione **Core Data Services > Metadata Extension**.
3. Nomeie o objeto (ex: Z\_C\_TRAVEL\_M).

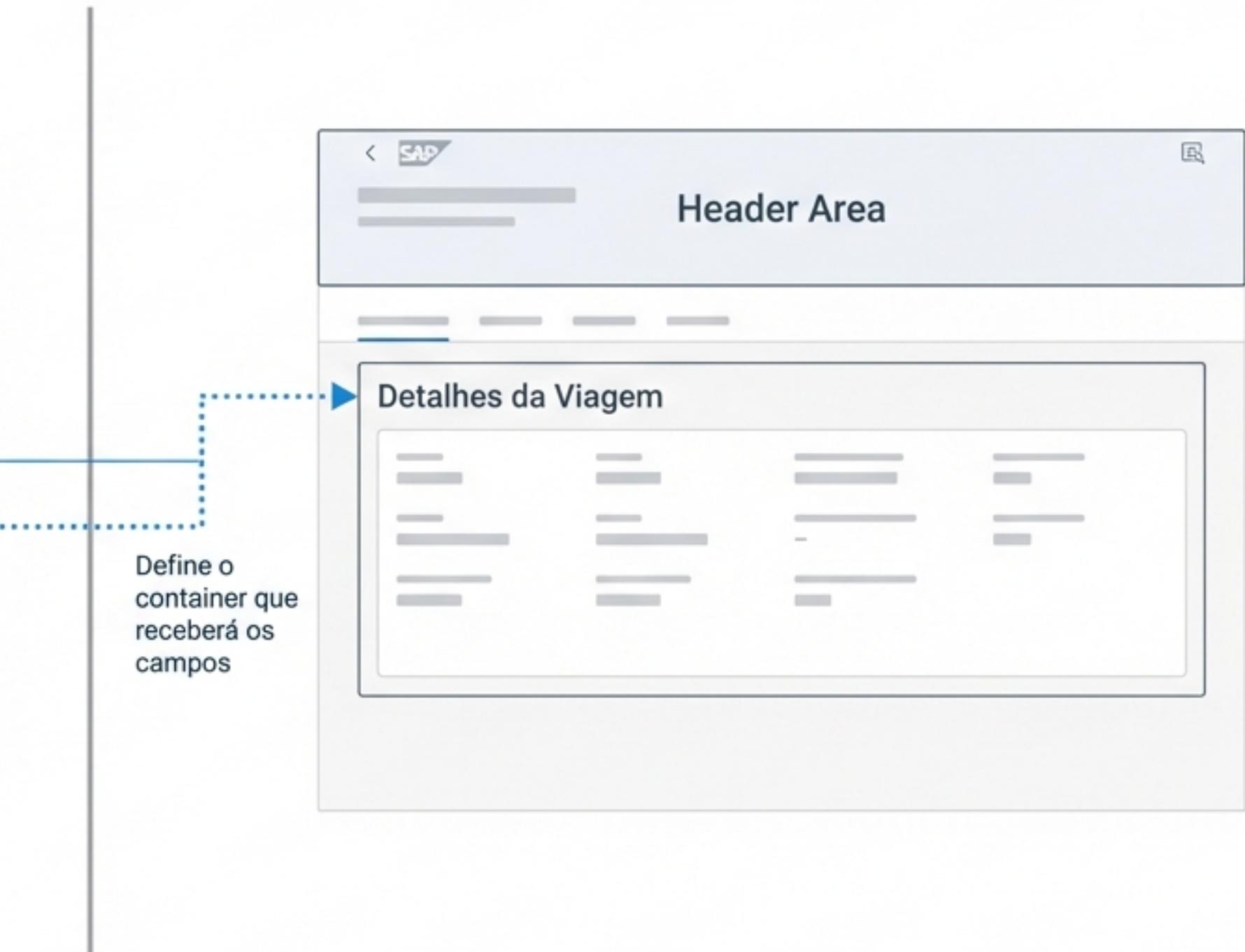
```
@Metadata.layer: #CORE  
annotate view Z_C_TRAVEL with  
{  
    /* As configurações de campo  
    virão aqui */  
}
```



Vínculo com  
a CDS View  
original

# Estruturando a Página: Header e Facets

```
@UI.facet: [  
  {  
    id: 'Travel',  
    purpose: #STANDARD,  
    type: #IDENTIFICATION_REFERENCE,  
    label: 'Detalhes da Viagem',  
    position: 10  
  }  
]
```



# Detalhando Campos: Colunas, Filtros e Cores

```
/* TravelID: Primeira coluna e primeiro filtro */
@UI: { lineItem: [{ position: 10, importance: #HIGH }],
        selectionField: [{ position: 10 }] }
TravelID;

/* Status: Cor dinâmica (0-3) */
@UI.lineItem: [{ position: 50, criticality: 'StatusCriticality' }]
OverallStatus;

/* CustomerID: Ocultável em mobile */
@UI.lineItem: [{ position: 30, importance: #MEDIUM }]
CustomerID;
```



## lineItem

Define a ordem das colunas na tabela.



## selectionField

Cria o campo na barra de filtros superior.



## importance

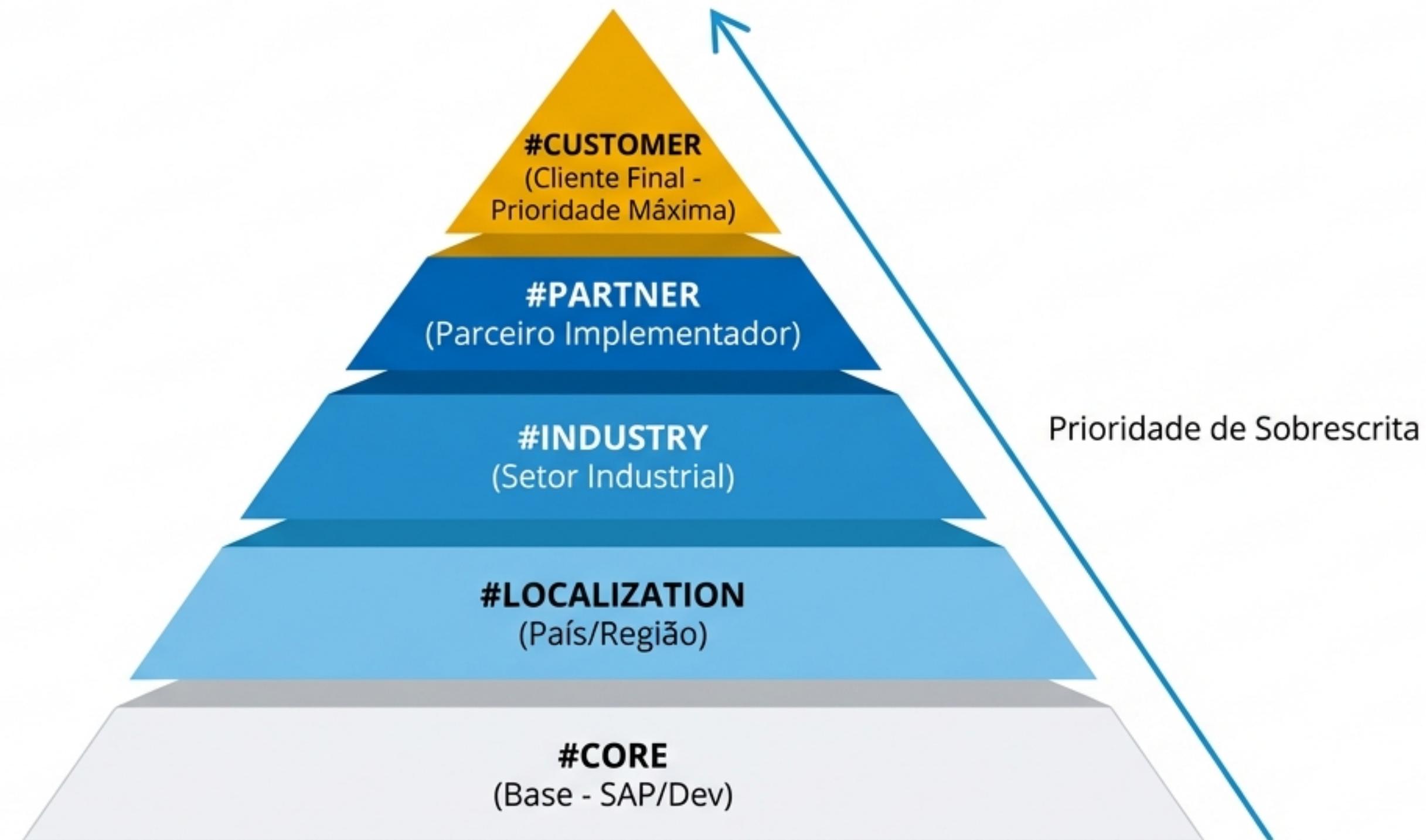
#HIGH (Sempre visível) vs #MEDIUM (Oculto em telas pequenas/mobile).



## criticality

Renderiza cores de semáforo (Verde/Amarelo/Vermelho) baseado no valor do campo apontado.

# O Poder das Camadas (Layering)



O sistema faz o 'merge' em tempo de execução. A camada superior sempre vence.

# Layering na Prática: Quem Vence?



Flexibilidade total sem tocar no código fonte original (Upgrade Safe).

# Comparativo: Inline vs. Metadata Extension

Característica	Inline (.ddls)	Extension (.ddlx)
Localização	Misturada ao SQL	Arquivo separado dedicado
Legibilidade	Baixa (Polui a lógica)	Alta (Foco em UI)
Flexibilidade	Rígida (Exige regeneração)	Flexível (Suporta camadas)
Uso Recomendado	Anotações Semânticas/Técnicas	Anotações de UI ( <code>@UI</code> )

# Glossário Técnico

**Metadata Extension (MDE):** Objeto .ddl que separa anotações de UI da definição de dados.

**Separation of Concerns:** Princípio onde o modelo de dados é agnóstico à sua apresentação.

**@Metadata.allowExtensions:** A ‘chave mestre’ na CDS View que autoriza o enriquecimento externo.

**Layer:** Propriedade de prioridade (#CORE -> #CUSTOMER) que gerencia conflitos de configuração.

**Criticality:** Propriedade de UI para renderização dinâmica de status (cores/ícones).

# Pontos de Atenção e Melhores Práticas



**DO:** Use MDE para **\*todas\*** as anotações relacionadas a UI (`@UI.*``).

```
Fira Code UI = (`@UI.*`, `Review`)
```



**DON'T:** Esquecer `@Metadata.allowExtensions: true` (Causa tela sem formatação).

```
@Metadata.allowExtensions: true
```



**DO:** Utilize a propriedade `'importance'` para garantir que campos críticos apareçam em dispositivos móveis.

```
Fira Code paradel importance=status
```



**INSIGHT:** 'Clean Code' não é apenas sobre lógica eficiente, é sobre organização e responsabilidade única dos arquivos.

# Quiz de Fixação

**Q1:** O que acontece com a UI se o arquivo MDE estiver correto, mas a CDS View não tiver

```
allowExtensions:  
true?
```

**Q2:** Por que o uso de MDE promove o princípio de "Clean Code" no RAP?

**Q3:** Em um conflito de anotações entre a camada `#CORE` e a camada `#CUSTOMER`, qual configuração prevalece na tela?

# **Transforme seu código: Dados no Backend, Estilo na Extensão.**

Revise sua última CDS View e refatore as anotações de UI para um arquivo MDE hoje mesmo.