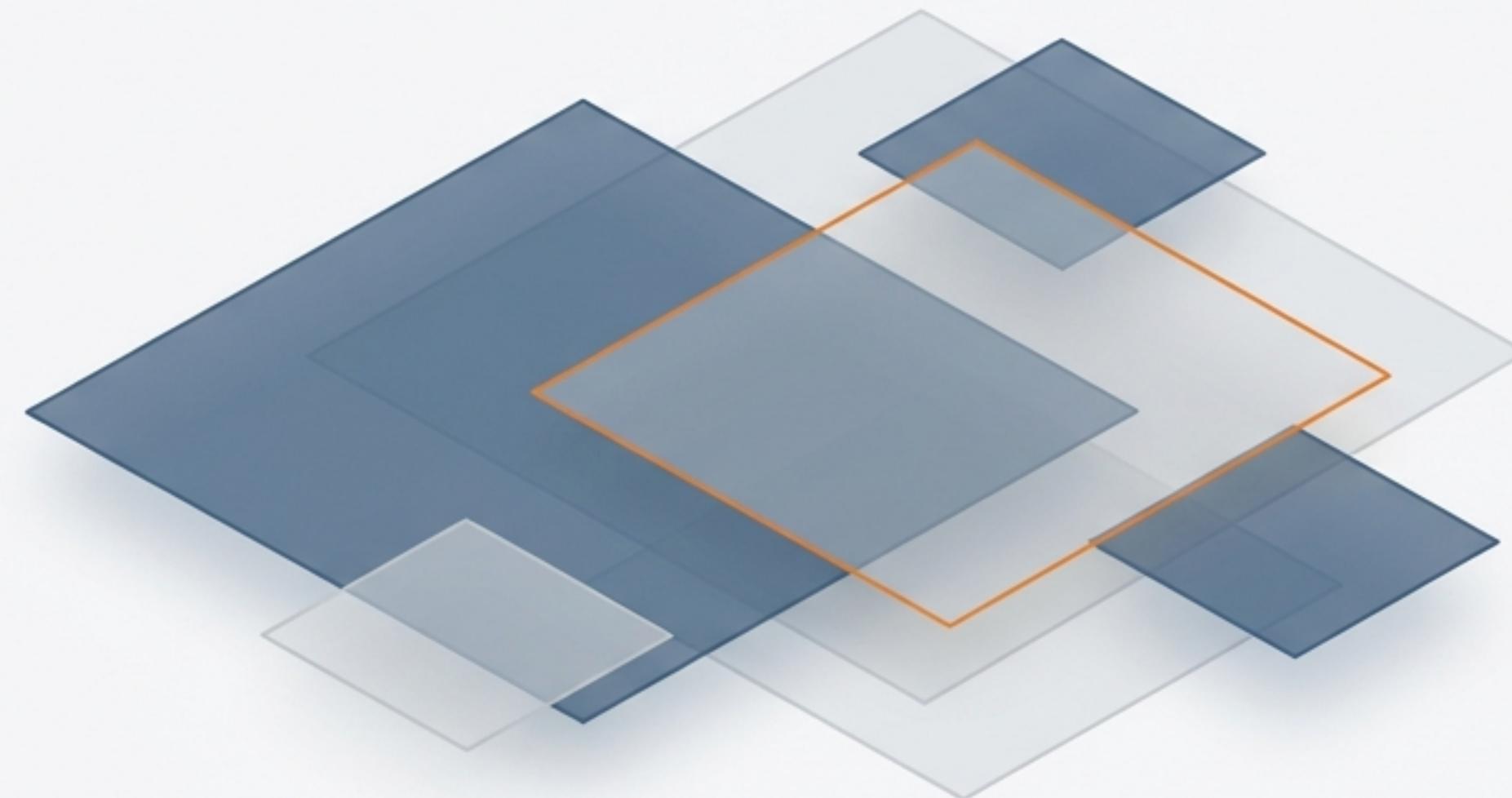


# Desvendando 'Localized Data' no CAP

Da Simplicidade Declarativa à Mecânica do Runtime



# O Princípio Fundamental: Capture a Intenção, Não o "Como"

## A Maneira CAP

### A Intenção

```
// Modelo de domínio limpo, focado no negócio.  
entity Books : cuid, managed {  
    title : localized String;  
    descr : localized String;  
    // ...  
}
```

Com o CAP, você declara a *intenção* de que um campo seja traduzível. O framework cuida da complexidade.

## A Maneira Manual

### O "Como"

```
// Modelo poluído com detalhes de implementação.  
entity Books {  
    key ID : UUID;  
    title : String; // Idioma padrão  
    descr : String; // Idioma padrão  
    texts : Composition of many Books_texts on texts.book = $self;  
}  
  
entity Books_texts {  
    key book : Association to Books;  
    key locale : String;  
    title : String;  
    descr : String;  
}
```

A abordagem manual exige entidades de texto separadas, chaves estrangeiras complexas e lógica de join, poluindo o modelo de domínio principal.

# A Superfície: A Simplicidade do Modificador `localized`

Para tornar um campo traduzível, basta adicionar o modificador `localized` ao seu tipo. O modelo de domínio permanece limpo e focado no negócio.

```
using { cuid, managed } from '@sap/cds/common';

entity Books : cuid, managed {
    title    : localized String(111);
    descr    : localized String(1111);
    author   : Association to Authors;
    stock    : Integer;
}
```

## Restrições

O modificador `localized` não pode ser usado em chaves de entidade que sejam associações, nem em subelementos de tipos estruturados.

# Nos Bastidores (Parte 1): A Mágica do Compilador

A simplicidade do `localized` esconde um trabalho pesado realizado pelo compilador CDS. Ele gera automaticamente os artefatos necessários para armazenar e acessar as traduções.

## 1. Geração da Entidade de Textos

Para cada entidade com campos `localized`, uma entidade `.texts` correspondente é criada. Ela contém as chaves da entidade original mais uma coluna `locale` para armazenar as traduções.

```
entity Books {  
    title : localized String;  
    ...  
}
```

```
entity Books.texts {  
    key locale : sap.common.Locale;  
    key ID      : UUID; // Chave da entidade original  
    title     : String(111);  
    descr     : String(1111);  
}
```

## 2. Extensão da Entidade Base

A entidade original é estendida com duas associações que conectam os textos traduzidos:

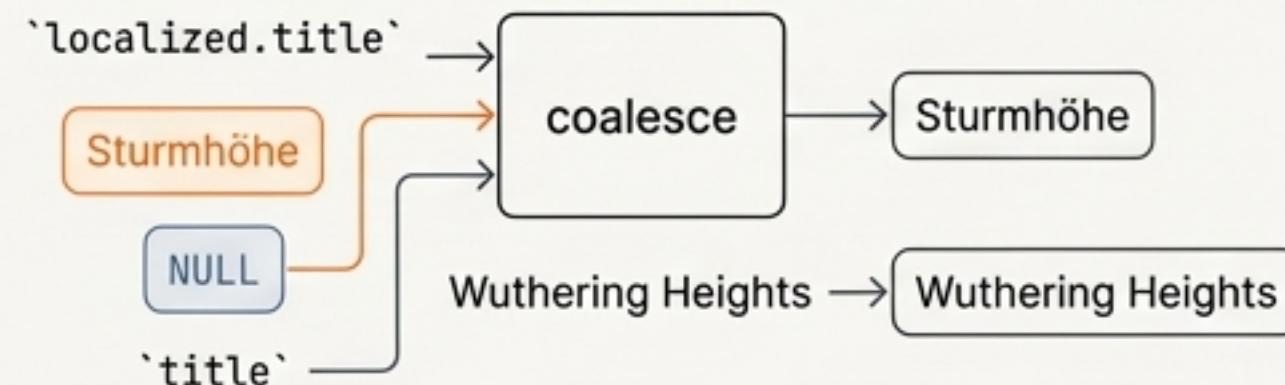
- `texts`: Uma composição para acessar *\*todas\** as traduções.
- `localized`: Uma associação especial que utiliza **\$user.locale** para buscar a tradução correta\* para o usuário atual.

```
extend entity Books with {  
    texts   : Composition of many Books.texts on texts.ID = ID;  
    localized : Association to Books.texts on localized.ID = ID  
        and localized.locale = $user.locale;  
}
```

# Nos Bastidores (Parte 2): Acesso Transparente via Views SQL

Além das entidades, o compilador gera uma view SQL especial, prefixada com `localized.`, que se torna o ponto de acesso padrão para leituras. Esta view implementa a lógica de fallback de idioma de forma transparente.

```
-- A view `localized.Books` é usada por padrão em requisições de leitura.  
entity localized.Books as select from Books {*,  
    coalesce(localized.title, title) as title,  
    coalesce(localized.descr, descr) as descr  
};
```



## A Função `coalesce`

`coalesce` retorna o primeiro valor não nulo da lista. Neste caso, ele tenta buscar o texto da associação `localized` (que já está filtrada pelo idioma do usuário). Se não encontrar, ele utiliza o valor da própria entidade `Books`, que serve como idioma padrão.

O resultado? Uma consulta `SELECT title from Books` magicamente retorna o texto no idioma do usuário ou o texto padrão, sem esforço adicional.

# O Motor do Runtime: O Papel Essencial do `'\$user.locale`'

A "mágica" de selecionar o idioma correto é controlada pela pseudovariável `'\$user.locale'`. O runtime do CAP define essa variável com base na requisição do usuário.



**Requisição HTTP:** Um cliente (navegador) envia uma requisição com o header `Accept-Language: de-DE`.

**Runtime CAP:** O framework CAP inspeciona o header e determina o locale preferido do usuário.

**Definição da Variável:** O runtime define internamente `$user.locale = 'de'.`

**Consulta ao Banco:** A associação `localized` na view `localized.Books` utiliza esta variável para fazer o join com a tradução correta na tabela `Books.texts`.

O `$user.locale` é a ponte entre o contexto do usuário e a seleção de dados, garantindo que a tradução correta seja entregue de forma automática e eficiente.

# Lendo Dados: Dois Cenários, Um Modelo

O modelo `localized` do CAP serve de forma elegante a diferentes casos de uso de leitura, sem a necessidade de criar serviços separados.

## Para o Usuário Final (Leitura Transparente)

**Use Case:** Exibir o título de um livro em uma aplicação web.

### How it Works

Uma consulta simples na entidade `Books` utiliza automaticamente a view `localized.Books` nos bastidores.

```
SELECT ID, title from Books WHERE ID = 201;  
-- Se $user.locale = 'de', retorna 'Sturmhöhe'.  
-- Se $user.locale = 'fr', retorna 'Les Hauts de Hurlevent'.  
-- Se a tradução não existir, retorna 'Wuthering Heights'.
```

### Result

O usuário recebe o conteúdo no seu idioma, com fallback automático.

## Para UIs de Tradução (Acesso Direto)

**Use Case:** Uma tela administrativa para editar todas as traduções de um livro.

### How it Works

Acessa-se diretamente a composição `texts`, que não é afetada pelo `'\$user.locale'`.

```
GET /AdminService/Books(201)/texts
```

### Result

A API retorna um array com todas as traduções disponíveis, permitindo a edição completa.

```
[  
  {"locale": "de", "title": "Sturmhöhe", ...},  
  {"locale": "fr", "title": "Les Hauts de Hurlevent", ...}  
]
```

# Escrevendo Dados: Criando e Atualizando Traduções

O modelo gerado é totalmente compatível com as operações de escrita padrão do OData. Você pode criar, adicionar, atualizar e remover traduções através da API.

## 1. Criar uma Entidade com Traduções (Deep Insert)

Ao criar um novo livro, você pode fornecer as traduções iniciais no mesmo payload, dentro do array 'texts'.

```
POST /AdminService/Books  
Content-Type: application/json  
  
{  
  "ID": "...",  
  "title": "Some new book",  
  "texts": [  
    { "locale": "de", "title": "Ein neues Buch" },  
    { "locale": "fr", "title": "Un nouveau livre" }  
  ]  
}
```

## 2. Adicionar uma Nova Tradução a uma Entidade Existente

Faça um 'POST' diretamente na composição 'texts' da entidade desejada.

```
POST /AdminService/Books(ID...)/texts  
Content-Type: application/json  
  
{ "locale": "es", "title": "Un libro nuevo" }
```

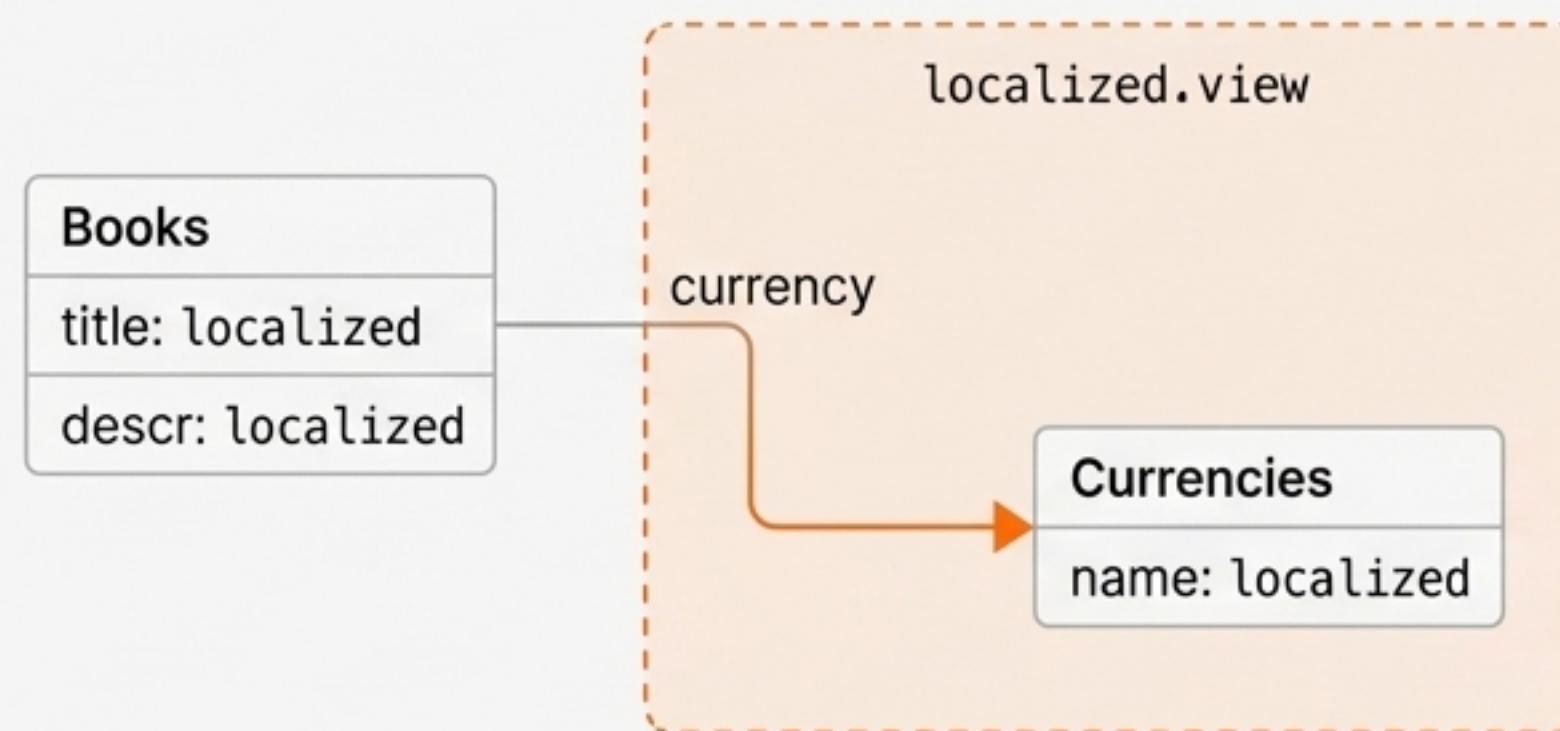
## 3. Atualizar ou Remover uma Tradução Específica

Utilize 'PUT', 'PATCH' ou 'DELETE' no endpoint específico da tradução, identificando-a pela chave composta (ID da entidade + locale).

```
PATCH /AdminService/Books(ID...)/texts(ID=..., locale='de')  
DELETE /AdminService/Books(ID...)/texts(ID=..., locale='de')
```

# Tópico Avançado: Dados Localizados Aninhados

O poder do `localized` se estende através das associações. Se uma entidade `A` tem uma associação para uma entidade `B` que também possui dados localizados, o CAP resolve ambas as traduções de forma transparente.



## A Mágica do Redirecionamento Automático

Quando o compilador cria a view `localized.Books`, ele detecta que a associação `currency` aponta para uma entidade também localizada e a redireciona automaticamente para a view `localized.Currencies`.

```
entity localized.Books as select from Books AS p mixin {  
    // A associação é redirecionada para a view `localized.`  
    currency : Association to localized.Currencies on currency =  
        currency = p.currency;  
}  
    } into {* /*...*/};
```

## Resultado Final

Uma única consulta que expande a associação de moeda retornará tanto o título do livro quanto o nome da moeda traduzidos para o idioma do usuário.

```
SELECT from localized.Books {  
    ID, title, descr,  
    currency.name as currency  
}
```

# Tópico Prático: Populando o Banco com Dados Iniciais

Para carregar dados iniciais e suas traduções, você precisa de dois arquivos CSV: um para os dados no idioma padrão e outro para os textos traduzidos.

## `my.bookshop-Books.csv`

Contém os dados principais da entidade. Os campos `localized` (como `title` e `descr`) são preenchidos com o texto no idioma padrão da aplicação.

```
ID;title;descr;author_ID;stock;price;currency_code .....  
201;Wuthering Heights;"Wuthering Heights, Emily Brontë's only novel...";101  
207;Jane Eyre;"Jane Eyre is a novel by English writer...";107;11;12.34;GBP
```

## `my.bookshop-Books_texts.csv`

Contém apenas as traduções. Note o sufixo `\_texts`. As colunas são a chave da entidade original (`ID`), a coluna `locale`, e os campos traduzidos.

```
ID;locale:title;descr  
201;de;Sturmhöhe;"Sturmhöhe (Originaltitel: Wuthering Heights) ist der einzige  
201;fr;Les Hauts de Hurlevent;"Les Hauts de Hurlevent (titre original : Wuther.  
207;de;Jane Eyre;"Jane Eyre. Eine Autobiographie (Originaltitel: Jane Eyre. An.
```

**Nota Importante:** O nome do arquivo CSV para as traduções deve seguir o padrão `<namespace>-<entidade>\_texts.csv`.

# A História Completa: Simplicidade e Poder em Perfeita Harmonia



Escreve title: localized  
String;

Gera `Books.texts`,  
estende `Books` com  
associações e cria a view  
`localized.Books`.

Recebe a requisição HTTP,  
extrai o idioma do header  
`Accept-Language` e define  
`$user.locale`.

A consulta à view  
`localized.Books` utiliza a  
associação `localized` e  
`coalesce` para retornar o  
texto correto.



**Modelos de Domínio Limpos:**  
Foco total na lógica de negócio, livre de  
detalhes de implementação de i18n.



**Separação de Preocupações:**  
O modelo de domínio, a estrutura de  
persistência e a lógica de acesso são mantidos  
separados.



**Implementação Otimizada:**  
A lógica de fallback e acesso é genérica,  
robusta e otimizada pelo framework CAP.



**Redução de Código Boilerplate:**  
Elimina a necessidade de escrever código  
repetitivo e propenso a erros para gerenciar  
traduções.