Workshop Sofia from 04.08. - 08.08.2025

Workshop Series

The world is how we shape it

sopra steria

Q: Where are we and where are we going?

- *Yesterday* (previous workshop): You developed UI5 frontend - beautiful user interfaces with lists, dialogues and fragments
- **Today:** The path to real backend services that supply these UI5 apps with real data
- Tomorrow: static UI5 apps become dynamic applications with real data from SAP systems

C2 - Restricted use

Day 1

Retrospective: Brief recap of UI5 concepts from the previous workshop

Day 1 - Retrospective: Brief recap of UI5 concepts from the previous workshop

Workshop Sofia (04.08.2025)

- Q: What did we achieve in the previous workshop? (detailed recap)
  - **Nested Views** - Complex UI Structures

    Example:

```javascript
// Das haben Sie gestern gelernt:
// Nested Views = "Verschachtelte Ansichten"
sap.ui.core.mvc.View.create({
    viewName: "myapp.view.Main",
    type: "XML"
}).then(function(oView) {
    // Hier hatten Sie mehrere Views ineinander verschachtelt
    // Wie russische Puppen - eine View in der anderen
});
```


sopra steria

C2 - Restricted use

Day 1 - Retrospective: Brief recap of UI5 concepts from the previous workshop

Workshop Sofia (08.08.2025)

- Q: What was the goal?
  - Modular development: Each view had a specific task
  - Reusability: Views could be used in different contexts
  - Clarity: complex UIs were broken down into manageable parts

  Example:

```xml
<!-- MainView.view.xml -->
<mvc:View>
    <Page title="Rechnungsübersicht">
        <!-- Hier wurde eine NestedView eingebettet -->
        <mvc:XMLView viewName="myapp.view.InvoiceList" />
        <mvc:XMLView viewName="myapp.view.CustomerInfo" />
    </Page>
</mvc:View>
```


C2 - Restricted use

Day 1 - Retrospective: Brief recap of UI5 concepts from the previous workshop

Workshop Sofia (04.08.2025)

- **Dialogues** - user interaction
  - Q: What were dialogues in the previous workshop?Modular development: Each view had a specific task

    Example:

```javascript
// Dialog-Erstellung (was Sie gestern gemacht haben)
var oDialog = new sap.m.Dialog({
    title: "Neue Rechnung erstellen",
    content: [
        new sap.m.Input({placeholder: "Kundennummer"}),
        new sap.m.Input({placeholder: "Betrag"})
    ],
    buttons: [
        new sap.m.Button({
            text: "Speichern",
            press: function() {
                // HIER WAR DAS PROBLEM gestern:
                // Keine echten Daten - nur Simulation!
                sap.m.MessageToast.show("Rechnung erstellt (aber nicht
wirklich gespeichert)");
            }
        })
    ]
});
```


C2 - Restricted use

Day 1 - Retrospective: Brief recap of UI5 concepts from the previous workshop

Workshop Sofia (04.08.2025)

- The problem we are solving:
  - *Yesterday:* Dialogues only showed messages, but didn't save anything
  - **Today:** Dialogues are connected to real backend services
  - Tomorrow: When you click "Save", data is really saved in the database
- **Fragments** - reusable UI components - *Example:*

```xml
<!-- CustomerInfoFragment.fragment.xml (was Sie gestern erstellt haben) -->
<core:FragmentDefinition
    xmlns="sap.m"
    xmlns:core="sap.ui.core">

    <VBox>
        <Text text="Kundenname: Max Mustermann" />
        <Text text="Kundennummer: CUST-001" />
        <Text text="Status: Aktiv" />
    </VBox>
</core:FragmentDefinition>
```


C2 - Restricted use

Day 1 - Retrospective: Brief recap of UI5 concepts from the previous workshop

Workshop Sofia (04.08.2025)

- The problem:
  - All data was "hardcoded" (written in code)
  - Max Mustermann was always the same customer
  - CUST-001 never changed

The solution - *Example:*

```xml
<!-- Heute werden wir lernen, wie Fragments dynamisch werden -->
<VBox>
    <Text text="Kundenname: {customer/name}" />          <!-- ← Aus Backend -->
    <Text text="Kundennummer: {customer/number}" />      <!-- ← Aus Backend -->
    <Text text="Status: {customer/status}" />            <!-- ← Aus Backend -->
</VBox>
```


C2 - Restricted use

Day 1 - Retrospective: Brief recap of UI5 concepts from the previous workshop

Workshop Sofia (04.08.2025)

- The critical transition: From static to dynamic
  - Q: What was missing *yesterday*?
  - 1. **Real data** - *Example:*

```javascript
// GESTERN (statische Daten):
var aInvoices = [
    {id: 1, customer: "Max Mustermann", amount: "1.500 €"},
    {id: 2, customer: "Tech AG", amount: "2.300 €"}
];

// HEUTE (dynamische Daten aus Backend):
var oModel = new sap.ui.model.odata.v4.ODataModel({
    serviceUrl: "/sap/opu/odata/sap/ZINVOICE_SRV/"
});
// Daten kommen aus echten SAP-Systemen!
```


C2 - Restricted use

Day 1 - Retrospective: Brief recap of UI5 concepts from the previous workshop

Workshop Sofia (04.08.2025)

- The critical transition: From static to dynamic
  - Q: What was missing *yesterday*?
  - 2. **Real interactions** - *Example:*

```javascript
// GESTERN (nur Meldungen):
onSaveInvoice: function() {
    sap.m.MessageToast.show("Rechnung gespeichert (nicht wirklich)");
}

// HEUTE (echte Speicherung):
onSaveInvoice: function() {
    this.getModel().create("/Invoices", oInvoiceData)
        .then(() => sap.m.MessageToast.show("Rechnung wirklich gespeichert!"));
}
```


C2 - Restricted use

Day 1 - Retrospective: Brief recap of UI5 concepts from the previous workshop

Workshop Sofia (04.08.2025)

- The critical transition: From static to dynamic
  - Q: What was missing *yesterday*?
  - 3. **Real navigation between data:** - *Example:*

```javascript
// GESTERN (Navigation nur zwischen Views):
this.getRouter().navTo("detail");

// HEUTE (Navigation zu echten Datensätzen):
this.getRouter().navTo("detail", {
    invoiceId: oSelectedItem.getBindingContext().getProperty("ID")
});
```


C2 - Restricted use

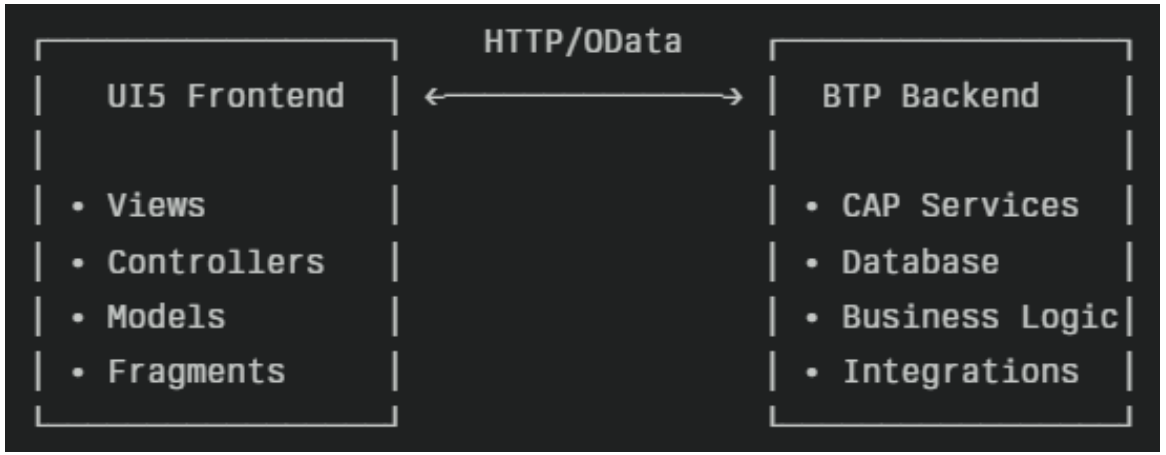
Day 1

Frontend-Backend Architecture: How UI5 apps communicate with backend services

Day 1 - Frontend-Backend Architecture: How UI5 apps communicate with backend services

Workshop Sofia (04.08.2025)

- Q: What is a frontend-backend architecture?
  - Simple analogy - *Example:*
    - Frontend (UI5) = restaurant (what guests see)
    - Backend (CAP/BTP) = kitchen (where the food is prepared)
    - API/OData = waiter (brings orders to the kitchen, food to guests)



C2 - Restricted use

Day 1 - Frontend-Backend Architecture: How UI5 apps communicate with backend services

Workshop Sofia (04.08.2025)

- Step-by-step communication process
  - Step 1: User clicks in UI5 – *Example:*

```javascript
// Benutzer klickt "Rechnungen laden" Button
onLoadInvoices: function() {
    // UI5 Controller wird ausgeführt
    console.log("Benutzer möchte Rechnungen sehen");
}
```


C2 - Restricted use

Day 1 - Frontend-Backend Architecture: How UI5 apps communicate with backend services

Workshop Sofia (04.08.2025)

- Step-by-step communication process
  - Step 2: UI5 sends request to backend - *Example:*

```javascript
onLoadInvoices: function() {
    // UI5 sendet HTTP-Request
    var oModel = this.getModel();
    oModel.read("/Invoices", {
        success: function(oData) {
            console.log("Backend hat geantwortet:", oData);
        }
    });
}
```


C2 - Restricted use

Day 1 - Frontend-Backend Architecture: How UI5 apps communicate with backend services

Workshop Sofia (04.08.2025)

- Step-by-step communication process
  - Step 3: Backend processes request - *Example:*

```javascript
// Backend (CAP Service) erhält Anfrage
// Liest Daten aus Datenbank
// Bereitet JSON-Response vor
{
    "d": {
        "results": [
            {"ID": "1", "CustomerName": "Max Mustermann", "Amount": 1500},
            {"ID": "2", "CustomerName": "Tech AG", "Amount": 2300}
        ]
    }
}
```


C2 - Restricted use

Day 1 - Frontend-Backend Architecture: How UI5 apps communicate with backend services

Workshop Sofia (04.08.2025)

- Step-by-step communication process
  - Step 4: UI5 displays data - *Example:*

```xml
<!-- UI5 Table wird automatisch mit Backend-Daten gefüllt -->
<Table items="{/Invoices}">
    <ColumnListItem>
        <Text text="{CustomerName}" />  <!-- ← Kommt vom Backend -->
        <Text text="{Amount}" />        <!-- ← Kommt vom Backend -->
    </ColumnListItem>
</Table>
```


C2 - Restricted use

Day 1

SAP BTP Overview:
Platform services for full-stack development

Day 1 – SAP BTP Overview: Platform Services für Full-Stack-Entwicklung

Workshop Sofia (04.08.2025)

- Q: What is SAP BTP and why do we need it?
  - BTP = Business Technology Platform (middleware, cloud-based)
  - Q: Why cloud? So that you don't have to worry about servers, updates and security
- BTP services that we will use **today**
  - 1. SAP **BAS** (**B**usiness **A**pplication **S**tudio):
    - Q: What is this? = Cloud-based code editor
    - Q: Why important? = You can develop from anywhere, everything is pre-installed
    - Analogy: = Like Google Docs, but for programming
  - 2. **CAP** (**C**loud **A**pplication **P**rogramming)
    - Q: What is this? = Framework for backend services
    - Q: Why important? = Automatically creates OData APIs for UI5
    - Analogy: = Like a construction kit for backend services
  - 3. SAP **HANA Cloud**
    - Q: What is this? = Cloud database
    - Q: Why is it important? = This is where your real data is stored
    - Analogy: = Like a huge, secure filing cabinet in the cloud

C2 - Restricted use

Day 1 – SAP BTP Overview: Platform Services für Full-Stack-Entwicklung

Workshop Sofia (04.08.2025)

- Understanding BTP architecture
  - **Global Account** (your company)
    - Subaccount 1 (Development)
      - Space: dev
      - Services: BAS, CAP, HANA
      - Apps Applications: Your apps
    - Subaccount 2 (Test)
    - Subaccount 3 (Production) BTP services that we will use **today**

  Explanation:
  - Global Account: Like your "house" at SAP
    - Subaccount: Like "rooms" in your house (one for development, one for testing, etc.)
      - Space: Like "areas" in a room
      - Services: Like "furniture" that you put in the rooms

C2 - Restricted use

Day 1

OData Services: The bridge between UI5 frontend and backend

Day 1 – OData Services: The bridge between UI5 frontend and backend

Workshop Sofia (04.08.2025)

- Q: What is OData and why is it so important?
  - **OData** = **O**pen **D**ata **P**rotocol
- Q: What does it do?
  - Standardised way for frontend and backend to talk to each other
- Q: Why is it important?
  - UI5 "understands" OData automatically

C2 - Restricted use

Day 1 – OData Services: The bridge between UI5 frontend and backend

Workshop Sofia (04.08.2025)

- OData vs. other APIs
  - Normal REST API call - *Example:*

```javascript
// Kompliziert - Sie müssen alles selbst programmieren
fetch('/api/customers')
    .then(response => response.json())
    .then(data => {
        // Sie müssen selbst programmieren:
        // - Daten-Bindung
        // - Filter
        // - Sortierung
        // - Paging
    });
```


sopra steria

C2 - Restricted use

Day 1 – OData Services: The bridge between UI5 frontend and backend

Workshop Sofia (04.08.2025)

- OData vs. other APIs
  - OData API Call (automatically with UI5) - *Example:*

```javascript
// Einfach - UI5 macht automatisch alles:
var oModel = new sap.ui.model.odata.v4.ODataModel({
    serviceUrl: "/odata/v4/invoice/"
});
this.setModel(oModel);

// UI5 macht automatisch:
// ✅ Daten-Bindung
// ✅ Filter
// ✅ Sortierung
// ✅ Paging
// ✅ Error Handling
```


C2 - Restricted use

Day 1 – OData Services: The bridge between UI5 frontend and backend

Workshop Sofia (04.08.2025)

- OData features for UI5 developers
  - 1. Automatic **data binding** - *Example:*

```xml
<!-- UI5 holt Daten automatisch vom OData-Service -->
<Table items="{/Invoices}">
    <ColumnListItem>
        <Text text="{InvoiceNumber}" />
        <Text text="{CustomerName}" />
        <Text text="{Amount}" />
    </ColumnListItem>
</Table>
```


C2 - Restricted use

Day 1 – OData Services: The bridge between UI5 frontend and backend

Workshop Sofia (04.08.2025)

- OData features for UI5 developers
  - 2. automatic **filters** - *Example:*

```javascript
// Benutzer tippt in Suchfeld - UI5 filtert automatisch
var oFilter = new sap.ui.model.Filter("CustomerName", "Contains", sSearchValue);
this.getView().byId("invoiceTable").getBinding("items").filter(oFilter);
// Backend bekommt automatisch: /Invoices?
$filter=contains(CustomerName,'searchvalue')
```

  - 3. automatic **sorting** - *Example:*

```javascript
// Benutzer klickt Spalten-Header - UI5 sortiert automatisch
var oSorter = new sap.ui.model.Sorter("Amount", false); // false = aufsteigend
this.getView().byId("invoiceTable").getBinding("items").sort(oSorter);
// Backend bekommt automatisch: /Invoices?$orderby=Amount asc
```


sopra steria

C2 - Restricted use

Day 1 – OData Services: The bridge between UI5 frontend and backend

Workshop Sofia (04.08.2025)

- OData metadata: How UI5 knows what backend can do
  - Metadata - *Example:*

```xml
<!-- Backend teilt UI5 mit: "Das sind meine Datenstrukturen" -->
<EntityType Name="Invoice">
    <Key>
        <PropertyRef Name="ID"/>
    </Key>
    <Property Name="ID" Type="Edm.Guid" Nullable="false"/>
    <Property Name="InvoiceNumber" Type="Edm.String" MaxLength="20"/>
    <Property Name="CustomerName" Type="Edm.String" MaxLength="100"/>
    <Property Name="Amount" Type="Edm.Decimal" Precision="15" Scale="2"/>
</EntityType>
```

UI5 automatically knows:

- ID is of type GUID (UUID)
- InvoiceNumber is string, max 20 characters
- Amount is decimal with 2 decimal places
- UI5 can automatically perform validation and formatting

C2 - Restricted use

Day 1

Subaccounts, spaces, and service architecture

Day 1 – Subaccounts, spaces, and service architecture

Workshop Sofia (04.08.2025)

- Subaccounts, spaces and service architecture
  - BTP structure for development teams
  - Development landscape:
- Workshop-BTP-Account

  DEV-Subaccount (Development)
  - Your personal space
  - Services: BAS, CAP Development
  - Test Database

  TEST subaccount (joint tests)
  - Integration Testing
  - Staging Database

  PROD subaccount (live system)
  - Productive Apps
  - Production Database

C2 - Restricted use

Day 1 – Subaccounts, spaces, and service architecture

Workshop Sofia (04.08.2025)

- Service architecture in BTP

  How services work together:

  - **Frontend-Tier** (UI5)
    - Fiori Launchpad
    - UI5 Applications
    - Custom UI5 Apps
  - **Application-Tier** (CAP)
    - Business Logic
    - OData Services
    - Custom APIs
  - **Integration-Tier**
    - Cloud Integration
    - API Management
    - Event Mesh
  - **Data-Tier**
    - SAP HANA Cloud
    - External Systems
    - Master Data

C2 - Restricted use

Day 1

Practical preparation:
From UI5 concepts to backend integration

Day 1 – Subaccounts, spaces, and service architecture

Workshop Sofia (04.08.2025)

- Practical preparation: From UI5 concepts to backend integration
  - Mindset change: From "mock" to "real"
  - *Yesterday* you thought - *Example:*

```javascript
// "Ich erstelle schöne UIs mit Beispieldaten"
var aCustomers = [
    {name: "Max Mustermann", id: "CUST-001"},
    {name: "Tech AG", id: "CUST-002"}
];
```

  - As of **today** you think - *Example*:

```javascript
// "Ich verbinde UIs mit echten Systemen"
var oModel = new sap.ui.model.odata.v4.ODataModel({
    serviceUrl: "/odata/v4/invoice/"  // ← Echte Daten aus echten Systemen
});
```


sopra steria

C2 - Restricted use

Day 1 – Subaccounts, spaces, and service architecture

Workshop Sofia (04.08.2025)

- Practical preparation: From UI5 concepts to backend integration
  - Q: What will change in your UI5 code?
    - 1. data binding becomes real - *Example:*

```xml
<!-- VORHER: Statische Daten -->
<Text text="Max Mustermann GmbH" />

<!-- NACHHER: Dynamische Daten -->
<Text text="{CustomerName}" />
```

    - 2. event handlers become functional - *Example:*:

```javascript
// VORHER: Nur Meldungen
onDeleteInvoice: function() {
    sap.m.MessageToast.show("Rechnung gelöscht (nur Simulation)");
}

// NACHHER: Echte Aktionen
onDeleteInvoice: function() {
    var oContext = this.getBindingContext();
    oContext.delete().then(() => {
        sap.m.MessageToast.show("Rechnung wirklich gelöscht");
    });
}
```


C2 - Restricted use

Day 1 – Subaccounts, spaces, and service architecture

Workshop Sofia (04.08.2025)

- Practical preparation: From UI5 concepts to backend integration
  - Q: What will change in your UI5 code?
    - 3. navigation becomes data-driven - *Example:*

```javascript
// VORHER: Feste Navigation
onShowDetails: function() {
    this.getRouter().navTo("details");
}

// NACHHER: Navigation mit echten IDs
onShowDetails: function(oEvent) {
    var oItem = oEvent.getSource();
    var sInvoiceId = oItem.getBindingContext().getProperty("ID");
    this.getRouter().navTo("details", {invoiceId: sInvoiceId});
}
```


C2 - Restricted use

Q&A