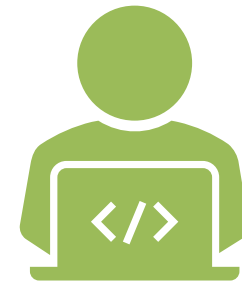


Web Applications and Tools Overview



- Overview of the technologies and tools used for developing our web applications.



- Focus on server, database, programming languages, and application functionalities.

Anmeldung

Anmeldung

Overview

Overview

Info

Bundle (jar)

Settings

Performance/Caching

Language/Compiler

Security

Regional

Charset

Scope

Request

Output

Error

Logging

Regex

Export

Services

Cache

Datasource

ORM

Mail

Tasks

Update

Restart

SSL Certificates

Extension

Applications

Providers

Archives & Resources

Mappings

Rest

Component

Custom tags

CFX tags

Debugging

Settings

Templates

Logs

Security

Access

Password

Services - Datasource

Settings

☐ Preserve single quotes ☐ Preserve single quotes (") in the SQL defined with the tag cfquery

update

cancel

Datasources

Datasources created here are available for ALL web contexts to use. The settings for these datasources can only be modified in this Server Administrator area.

<input type="checkbox"/>	Name	Type	Host/Server:Port	Open Connections	Storage	Check	
<input type="checkbox"/>	debbg0012a	Microsoft SQL Server (Vendor Microsoft)	debbg0012a.trench-group.net:1433	0	No		
<input type="checkbox"/>	loka	Microsoft SQL Server (Vendor Microsoft)	debbg0012a.trench-group.net:1433	0	No		

verify

cancel

delete

Create new datasource

Name

Type

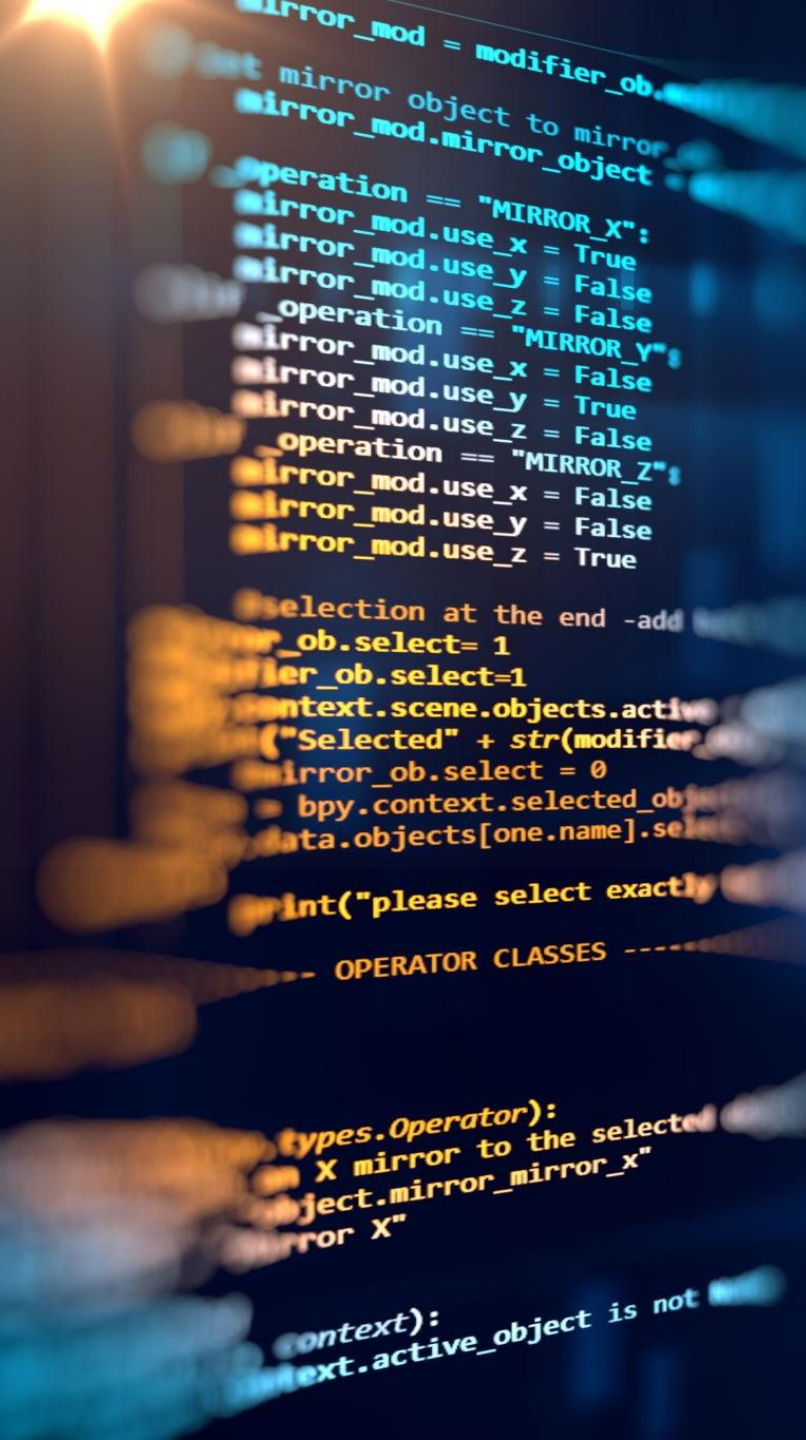
Missing the datasource type you are looking for? Click [here](#) to find an extension for the datasource type you require.

create

cancel

Server and Database Connectivity

- Server: <https://debbg0022a.trench-group.net/>
- Lucee Server: [Lucee 6.0.1.83 - Overview](#) - [Lucee Server Administrator](#)
- Open-source platform.
- Connections established with Microsoft SQL Server Database.



Programming Languages and Technologies

- - Programming Languages Used in Lucee:
- - HTML
- - CFML (ColdFusion Markup Language)
- - JavaScript
- - CSS
- - API
- - XML
- - SQL (for CFQUERY)
- - more...

Web Label APP Integration

- - Functionality:
- - Connection from web server to NiceLabel.
- - Execution of requests for preview and print of labels.
- - Connection from webserver to Microsoft SQL Database

```
function sendXMLForPreview() {
    const xmlData = document.getElementById('xmlContent').value;
    const myURL = 'https://debbg0013a.trench-group.net:50001/Webservice4XML NEW';

    const xmlhttp = new XMLHttpRequest();
    xmlhttp.responseType = 'text';

    xmlhttp.onreadystatechange = function() {
        if (xmlhttp.readyState === 4) {
            console.log('Response Type:', xmlhttp.responseType);
            if (xmlhttp.status === 200) {
                const response = parseXMLResponse(xmlhttp.responseText);

                // For preview action - show image
                const imageURL = "data:image/jpeg;base64," + response.preview64;

                // Displaying the image
                const previewImage = document.createElement('img');
                previewImage.src = imageURL;
                previewImage.alt = 'Preview Image';
                document.getElementById('result').innerHTML = '';
                document.getElementById('result').appendChild(previewImage);
                console.log('Response:', response.responseText);
            } else {
                console.error('Error: ', xmlhttp.status);
            }
        }
    };

    xmlhttp.open('POST', myURL, true);
    xmlhttp.setRequestHeader('Content-Type', 'text/plain');
    xmlhttp.send(xmlData);
}
```

```
// Constructing the XML string using the fetched data
xmlString = `<?xml version="1.0" standalone="no"?>
<!DOCTYPE labels SYSTEM "label.dtd">
<labels_FORMAT="${selectedFormat}" _JOBNAME="${gidValue}" _QUANTITY="${quantity}" _PRINTERNAME="${selectedPrinter}" _PREVIEW="TRUE" _PREVIEW_FORMAT="PNG">
  <label>
    <variable name="GID">${gidValue}</variable>
    <variable name="Zeile1">${escapeXml(Zeile1)}</variable>
    <variable name="Zeile2">${escapeXml(Zeile2)}</variable>
    <variable name="Zeile3">${escapeXml(Zeile3)}</variable>
  </label>
</labels>`;

xmlStringPrint = `<?xml version="1.0" standalone="no"?>
<!DOCTYPE labels SYSTEM "label.dtd">
<labels_FORMAT="${selectedFormat}" _JOBNAME="${gidValue}" _QUANTITY="${quantity}" _PRINTERNAME="${selectedPrinter}" _PRINT="TRUE">
  <label>
    <variable name="GID">${gidValue}</variable>
    <variable name="Zeile1">${escapeXml(Zeile1)}</variable>
    <variable name="Zeile2">${escapeXml(Zeile2)}</variable>
    <variable name="Zeile3">${escapeXml(Zeile3)}</variable>
  </label>
</labels>`;
```

Data Transfer and API Usage

- - Application: LOCA
- - SQL Queries: Interaction with the database.
- - API: Data transfer between our database and Thomas K's database.

PloP WEB Label APP LOCA

Linken Shipment Transformer Info

SerialNr: 3054805

Serial Number	Auftrag.Pos	Typ
3054655	2301000544.30	CTG420-040-005
3054804	2301000577.20	VTA420-01-001
3054654	2301000544.30	CTG420-040-005
3054805	2301000577.20	VTA420-01-001
3053236	2301000464.10	VCG123G05-001
3053242	2301000464.20	VCG123G05-002

SerialNr: 3054805
RFID: 20007AC941BC097F
Scan Dat: July, 12 2024 18:15:34 +02
Customer: Siemens Energy
Waren-Empfänger: TenneT TSO GmbH
Bezeichnung: SVA 420 - TenneT-Var.32
Feld: --> C2
Auftrag.Pos: 2301000577.20
Koordinaten: 49.8885683, 10.9155450
Kommentar:

```
authResponse = await fetch(`${base_url}/auth/login`, {
  method: 'POST',
  body: formData

authResponse.ok) {
  throw new Error('Authentication failed: ${authResponse.statusText}');

  authResult = await authResponse.json();
  accessToken = authResult.access_token;

  allRfidDataResponse = await fetch(`${base_url}/iot-data/rfid-data`, {
    method: 'GET',
    headers: {
      'Authorization': `Bearer ${accessToken}`
    }
  });

  if (!allRfidDataResponse.ok) {
    throw new Error('Failed to fetch RFID data: ${allRfidDataResponse.statusText}');
  }

  const allRfidData = await allRfidDataResponse.json();
  console.log('All RFID Data:', allRfidData);

  // Send RFID data to updateDatabase.cfm using FormData
  const formDataRfid = new FormData();
  formDataRfid.append("rfidData", JSON.stringify(allRfidData));

  const importDataResponse = await fetch('updateDatabase.cfm', {
    method: 'POST',
    body: formDataRfid // Send RFID data as FormData
  });
```

SQL Server Procedures

- - Procedures in Microsoft SQL Server Database:
- - Used for multiple reports.
- - Data analysis.
- - Table generation for subsequent use in WEB Label APP.

```
USE [uniBase]
GO
/***** Object: StoredProcedure [dbo].[ABC_ANALYZE_UPDATE_NEW]    Script Date: 07/15/2024 10:35:10 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =====
-- Author:      Ivelin Tanev
-- Create date: 11/10/2023
-- Description: ABC_ANALYZE_UPDATE
-- =====
ALTER PROCEDURE [dbo].[ABC_ANALYZE_UPDATE_NEW]

AS
BEGIN
```


Web-based Reporting System

- - New Development:
- - Creating a web-based report system.
- - Goal: Avoid using Alteryx and Excel exports.

MD04 Analyse

Abmelden

Filter by Material

Filter by MaterialKurzText

Filter by Kosten_Pro

Filter by Dispoel

Export to Excel

100001	PL-AUF	07	24-08	24-09	24-10	24-11	24-12	25-01	25-02	25-03	25-04
100001	W-BEST	35									
100002	15		700								
100002	AR-RES			-873			-1197	-1194			
100002	AR-RES_Accumulated			-873			-2070	-3264			
100002	Best_mit LT		700								
100002	BS-AVI		700								
100002	PL-AUF						3506.3	1194			
100002	W-BEST	1.7									
100003	15			700							
100003	AR-RES	6	-766	-766			-766	-766	-766	-1506	-2298
100003	AR-RES_Accumulated	1	-1537	-2303			-3069	-3835	-4601	-6107	-8405
100003	Best_mit LT		700								
100003	BS-AVI		700								
100003	PL-AUF						2101.7		3064	4544	766
100003	W-BEST	39.3									
100004	15			950.1							
100004	AR-RES	09	-2109	-1406	-1406					-2109	
100004	AR-RES_Accumulated	09	-4518	-5924	-7330					-9439	
100004	Best_mit LT	1.1			1123.7						



Questions and Answers

- - Open floor for questions and further discussion.