EVIDEN

Identity and Access Management

DirX Identity

Business User Interface Configuration Guide

Version 8.10.15, Edition April 2026

Table of Contents

Preface

This manual provides information about installing, configuring and customizing the DirX Identity Business User Interface.

DirX Identity Documentation Set

*Version 8.10.15 | Build 1932 | Date 2026-04-30 *

The DirX Identity document set consists of the following manuals:

- *DirX Identity Introduction*. Use this book to obtain a description of DirX Identity architecture and components.
- *DirX Identity Release Notes*. Use this book to understand the features and limitations of the current release. This document is shipped with the DirX Identity installation as the file **release-notes.pdf**.
- *DirX Identity History of Changes*. Use this book to understand the features of previous releases. This document is shipped with the DirX Identity installation as the file **history-of-changes.pdf**.
- *DirX Identity Tutorial*. Use this book to get familiar quickly with your DirX Identity installation.
- *DirX Identity Provisioning Administration Guide*. Use this book to obtain a description of DirX Identity provisioning architecture and components and to understand the basic tasks of DirX Identity provisioning administration using DirX Identity Manager.
- *DirX Identity Connectivity Administration Guide*. Use this book to obtain a description of DirX Identity connectivity architecture and components and to understand the basic tasks of DirX Identity connectivity administration using DirX Identity Manager.
- *DirX Identity User Interfaces Guide*. Use this book to obtain a description of the user interfaces provided with DirX Identity.
- *DirX Identity Application Development Guide*. Use this book to obtain information how to extend DirX Identity and to use the default applications.
- *DirX Identity Customization Guide*. Use this book to customize your DirX Identity environment.
- *DirX Identity Integration Framework*. Use this book to understand the DirX Identity framework and to obtain a description how to extend DirX Identity.
- *DirX Identity Web Center Reference*. Use this book to obtain reference information about the DirX Identity Web Center.
- *DirX Identity Web Center Customization Guide*. Use this book to obtain information how to customize the DirX Identity Web Center.
- *DirX Identity Meta Controller Reference*. Use this book to obtain reference information about the DirX Identity meta controller and its associated command-line programs and files.
- *DirX Identity Connectivity Reference*. Use this book to obtain reference information about the DirX Identity agent programs, scripts, and files.
- *DirX Identity Troubleshooting Guide*. Use this book to track down and solve problems in your DirX Identity installation.
- *DirX Identity Installation Guide*. Use this book to install DirX Identity.

- *DirX Identity Migration Guide*. Use this book to migrate from previous versions.

Notation Conventions

Boldface type
In command syntax, bold words and characters represent commands or keywords that must be entered exactly as shown.

In examples, bold words and characters represent user input.

Italic type
In command syntax, italic words and characters represent placeholders for information that you must supply.

[ ]
In command syntax, square braces enclose optional items.

{ }
In command syntax, braces enclose a list from which you must choose one item.

In Tcl syntax, you must actually type in the braces, which will appear in boldface type.

|
In command syntax, the vertical bar separates items in a list of choices.

...
In command syntax, ellipses indicate that the previous item can be repeated.

userID_home_directory
The exact name of the home directory. The default home directory is the home directory of the specified UNIX user, who is logged in on UNIX systems. In this manual, the home pathname is represented by the notation *userID_home_directory*.

install_path
The exact name of the root of the directory where DirX Identity programs and files are installed. The default installation directory is *userID_home_directory*/**DirX Identity** on UNIX systems and **C:\Program Files\DirX\Identity** on Windows systems. During installation the installation directory can be specified. In this manual, the installation-specific portion of pathnames is represented by the notation *install_path*.

dirx_install_path
The exact name of the root of the directory where DirX programs and files are installed. The default installation directory is *userID_home_directory*/**DirX** on UNIX systems and **C:\Program Files\DirX** on Windows systems. During installation the installation directory can be specified. In this manual, the installation-specific portion of pathname is represented by the notation *dirx_install_path*.

dxi_java_home
The exact name of the root directory of the Java environment for DirX Identity. This location is specified while installing the product. For details see the sections "Installation" and "The Java for DirX Identity".

tmp_path

The exact name of the tmp directory. The default tmp directory is /tmp on UNIX systems. In this manual, the tmp pathname is represented by the notation *tmp_path*.

tomcat_install_path
The exact name of the root of the directory where Apache Tomcat programs and files are installed. This location is defined during product installation.

mount_point
The mount point for DVD device (for example, **/cdrom/cdrom0**).

1. Installing and Configuring the DirX Identity Business User Interface

The DirX Identity Business User Interface is an application with a responsive Web design approach aimed at allowing Web page content to be viewed according to the size of the device being used to access the content. The application provides a fast and easy way to carry out approval tasks from a user's mobile device or desktop. The application communicates with DirX Identity services over the Internet through the HTTPS protocol.

The Business User Interface is available in the DirX Identity installation package and can be configured with the DirX Identity Configurator application. It requires the deployment of the DirX Identity REST Services.

This document describes the Business User Interface installation and configuration. It describes how to:

- Install the Business User Interface
- Configure the Business User Interface
- Customize the Business User Interface

1.1. Installing the Business User Interface

The DirX Identity Business User Interface is selected by default within the DirX Identity installation.However, you should make sure that this component is checked activated during installation, as shown in the following figure.

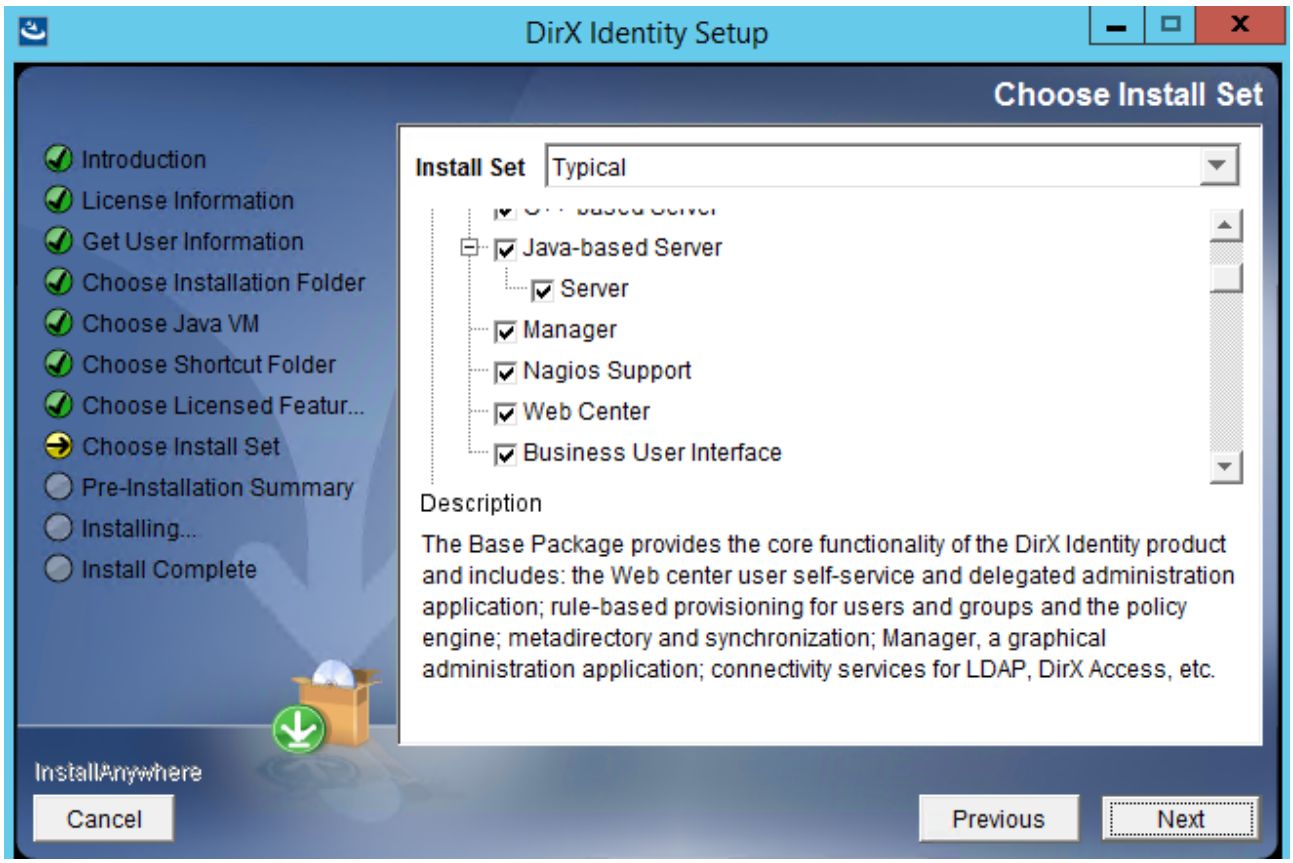

Figure 1. DirX Identity Installation

The Business User Interface requires the DirX Identity REST Services, which are installed by default.

The application is installed in the **web** subfolder of the DirX Identity installation path - next to the DirX Identity Web Center - as a template archive **BusinessUserInterface.zip** on Microsoft Windows or **BusinessUserInterface.tar** on Linux.This file is the base for the DirX Identity Configurator application to create different instances depending on the configured domain.

1.2. Deploying the Business User Interface

Use the DirX Identity Configurator application to configure the DirX Identity Business User Interface.You need to check the configuration option **Business User Interface Configuration**.You also need the **Identity Rest Services Configuration** to be configured as the interface to DirX Identity back end.

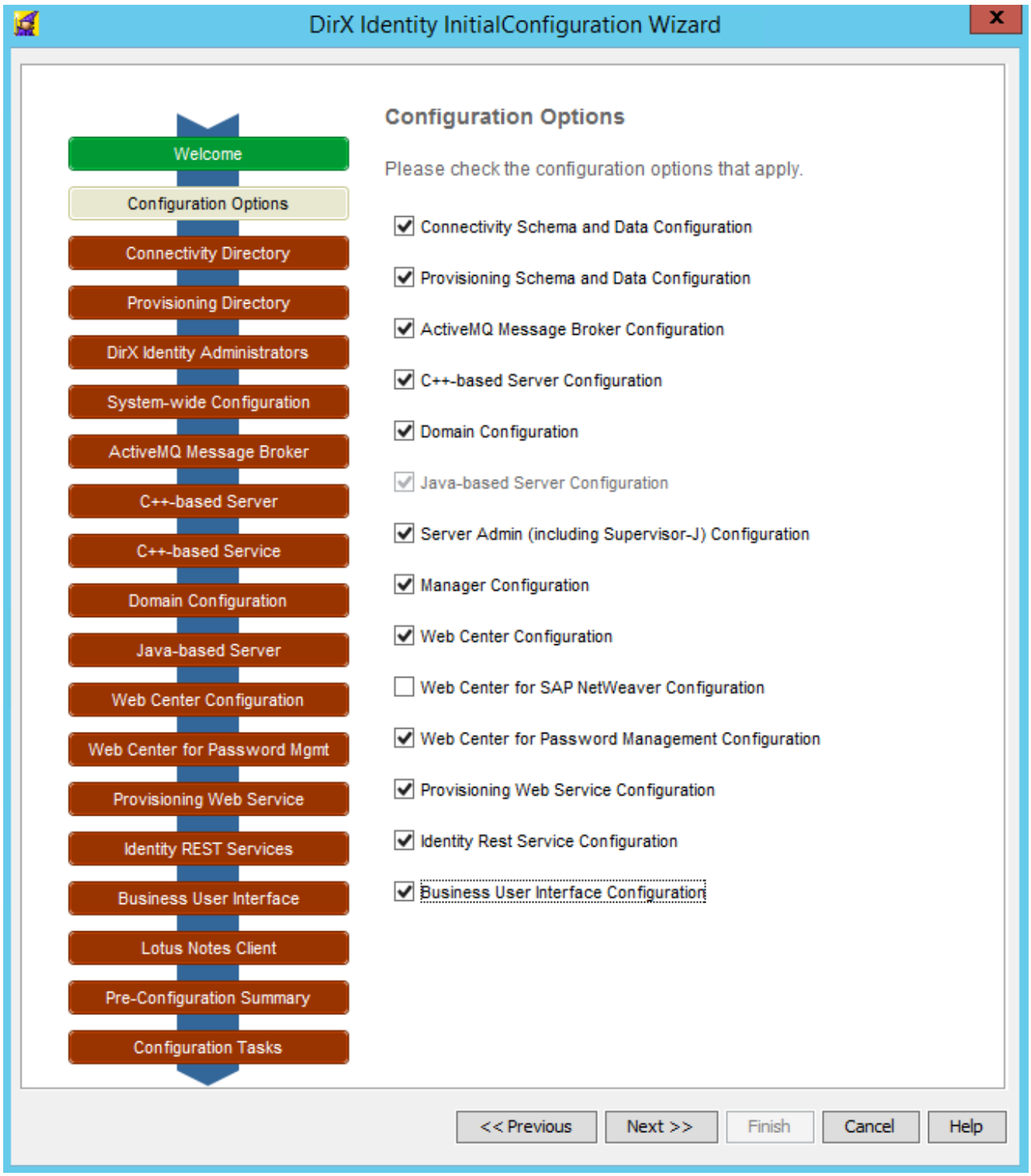

Figure 2. DirX Identity Configurator – Configuration Options

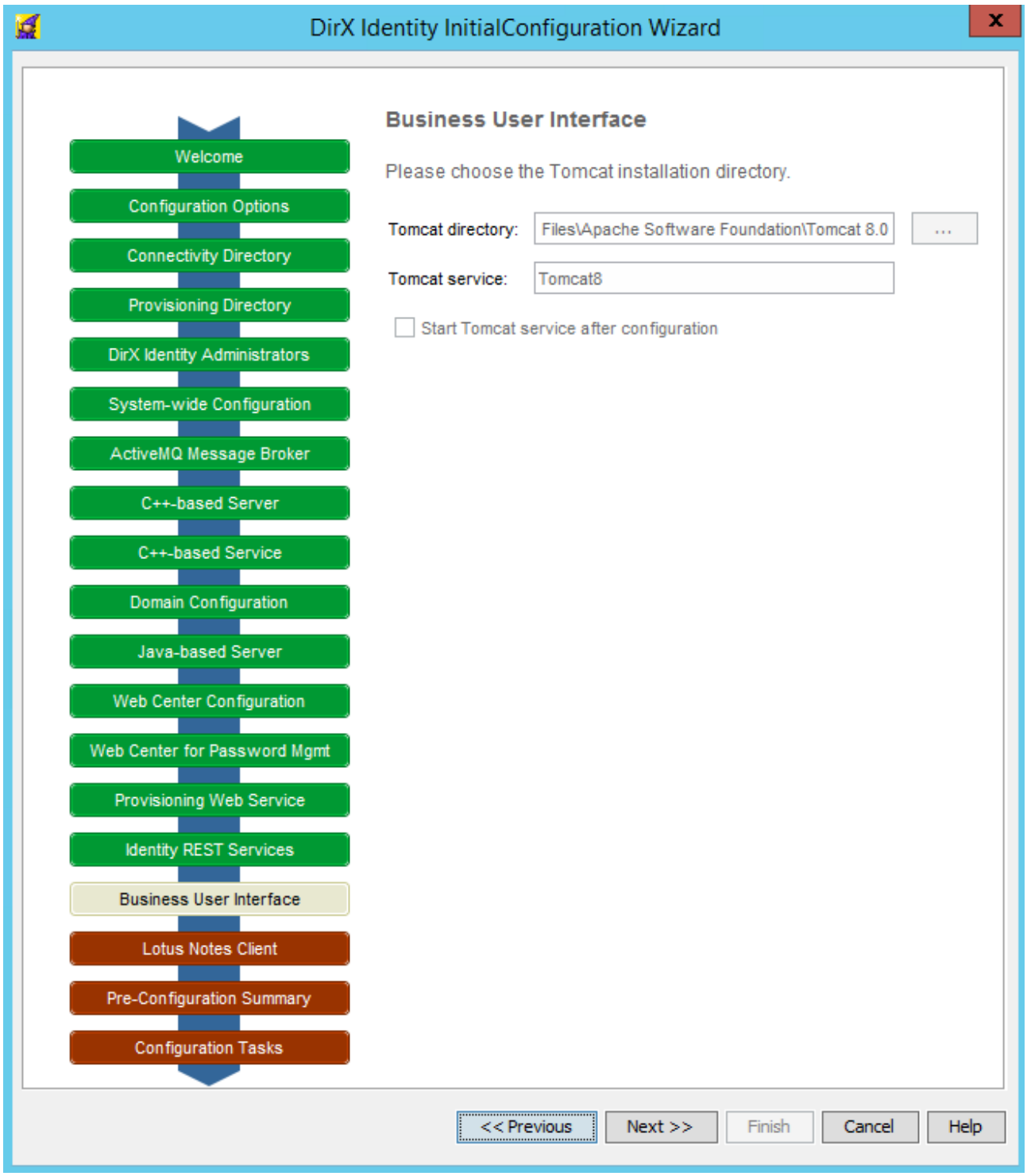

Figure 3. DirX Identity Configurator – Business User Interface Configurations

When you run the DirX Identity Configurator to install the DirX Identity Business User Interface, it creates a domain-specific deployment of the DirX Identity Business User Interface in the **web** folder of the DirX Identity installation folder.In our example, we configure the DirX Identity sample domain named **My-Company**.

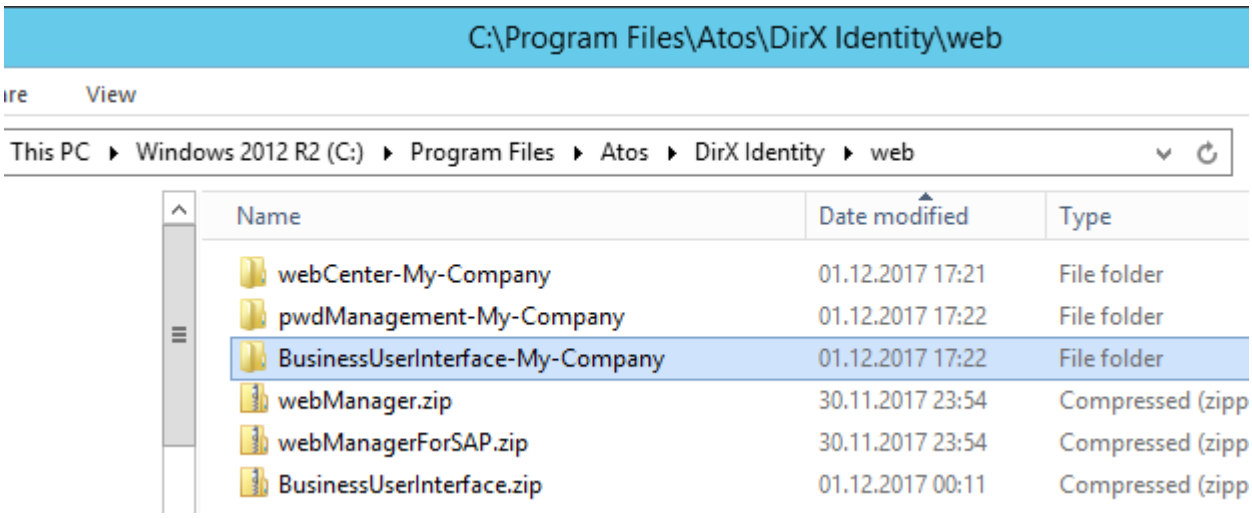

Figure 4. Business User Interface – Example Deployment

In addition, a domain-specific file called a context descriptor is deployed in the Tomcat server folder, which contains only the path to the Business User Interface deployment.The file is located at *tomcat_install_path***/conf/Catalina/localhost/BusinessUserInterface-My-Company.xml**.

The DirX Identity Configurator automatically configures the Business User Interface for using the DirX Identity REST Services, which will be deployed in the same Tomcat server.

If you want to use a Web server other than Tomcat, you can easily copy the folder **BusinessUserInterface-My-Company** from the **web** folder to the place where you need it.

1.3. Updating the Business User Interface with a Service Pack

It is good practice to back up the folder:

*install_path***/web/BusinessUserInterface-***domain*

particularly the **assets** subfolder - which contains your current configuration and customizations - to a safe place before installing a Service Pack or hotfix.After installing the Service Pack, you must start the configuration again as described in “Deploying the Business Unit Interface” to deploy the updated Business User Interface with the new features.This configuration step moves the previous installation of the Business User Interface to the folder:

*install_path***/web/BusinessUserInterface-***domain***.OLD**

which is overwritten by every successive configuration.After configuration, you need to re-customize your installation as described in the following sections.This task is easier if you merge your backed-up **assets** folder with the newly created one.

1.4. Configuring and Customizing the Business User Interface

Configuring the Business User Interface consists of the following tasks:

- Configuring access to the DirX Identity REST Services (mandatory)
- Disabling the risk feature (optional)
- Adding a new language (optional)
- Exchanging images and graphics (optional)

You can also make the following customizations to the Business User Interface:

- Customize tables using configuration files and/or user hooks
- Add new widgets to the default home page
- Configure access policies for BUI elements
- Customize user profiles using dynamic forms

The next sections describe these configuration and customization tasks.

1.4.1. Configuring Access to the DirX Identity REST Services

Configuration settings are available in the **config.json** file, located in the Business User Interface installation folder *install_path***/web/BusinessUserInterface-***domain***/assets/config/**. The content of this file is case-sensitive. The configuration includes:

“basicAuthServer”
: the connectivity URL to the DirX Identity REST Services for login with username and password. The username you must use is configured with the DirX Identity REST Services, not with the Business User Interface. For details, see the section "Configuring User Authentication" for the DirX Identity REST Services in the *DirX Identity Integration Framework Guide*. In the Business User Interface, you must configure the following parameters:

  “protocol”
  : the protocol, either **http** or **https**. If the Business User Interface is accessed through HTTPS, you can only use HTTPS to access the DirX Identity REST Services.

  “host”
  : the name of the server on which the DirX Identity REST Services is running. If you use HTTPS, you must use the same name as one of the Subject alternative names of the server certificate.

  “port”
  : the port on which the DirX Identity REST Services is accessible via HTTP or HTTPS. Configure **80** for the standard HTTP port. Configure **443** for the HTTPS standard port.

“domain”

the base path to the DirX Identity REST Services, which also configures the domain for which the Business User Interface is used.

The **domain** field contains the DirX Identity REST services context path and the DirX Identity domain; for example: **/DirXIdentityRestService-My-Company**. See the chapter “DirX Identity REST Services” in the *DirX Identity Integration Framework Guide* for more information.

“x509AuthServer”

- the connectivity URL to the DirX Identity REST Services for login with a PKI client certificate. You must configure the following parameters:

“protocol”

the protocol must be **https**.

“host”

the name of the server on which the DirX Identity REST Services is running. You must use the same name as one of the Subject alternative names of the server certificate.

“port”

the port on which the DirX Identity REST Services is accessible via HTTPS and on which the secure client authentication is configured on the Web server (Apache Tomcat).

“domain”

the base path to the DirX Identity REST Services, which also configures the domain for which the Business User Interface is used.

“kerberosAuthServer”

the connectivity URL to DirX Identity REST Services for login with Windows username and password. You must configure the following parameters:

“protocol”

the protocol must be **https**.

“host”

the name of the server on which the DirX Identity REST Services is running.

“port”

the port on which the DirX Identity REST Services is accessible via HTTPS and on which the secure client authentication is configured on the Web server (Apache Tomcat).

“domain”

the base path to the DirX Identity REST Services, which also configures the domain for which the Business User Interface is used.

“dxapepAuthServer”

the connectivity URL to DirX Identity REST Services for login with DirX Access Policy Enforcement Point (PEP). You must configure the following parameters:

“protocol”

the protocol must be **https**.

“host”

the name of the server on which the DirX Identity REST Services is running under DirX Access PEP configuration.

“port”

the port on which the DirX REST Services is accessible via HTTPS.

“domain”

the base path to the DirX Identity REST Services, which also configures the domain for which the Business User Interface is used.

These settings are for communication between Business User Interface and DirX Identity REST Services. Settings for DirX Access PEP are stored in the file:

tomcat_install_path/conf/server.xml*

Settings from **server.xml** are generated by DirX Access and must be transferred manually to Apache Tomcat.

Here is a Tomcat configuration snippet:

```
<Valve ClientId="Tomcat 9.0 - My-Company BUI"
       ClientKeystoreFilename=
"c:\Atos\Tomcat9PEP\conf\sec-comm.client.p12"
       ClientKeystorePassphrase="OfCHCMAQ10unQGnR"
       ServerPrimaryHost="my-server.my-company.example"
       ServerPrimaryPort="11111"
       ServerPrimaryPortSSL="11112"
       className="com.siemens.dxa.pep.tomcat9.DXAValve" />
</Engine>
<Connector SSLEnabled="true"
           acceptCount="100"
           clientAuth="want"
           debug="0"
           disableUploadTimeout="true"
           enableLookups="false"
           keystoreFile="conf/my-server.jks"
```

```
            keystorePass="dirxaccess"
            keystoreType="JKS"
            maxSpareThreads="75"
            maxThreads="150"
            minSpareThreads="25"
            port="8443"
            scheme="https"
            secure="true"
            sslProtocol="TLS"
            truststoreFile="conf/my-server.jks"
            truststorePass="dirxaccess"
            truststoreType="JKS" />
```

The most important parameters are:

“ClientId” - the name of the DirX Access PEP configuration.

“ServerPrimaryHost” - the DirX Access server name.

“ServerPrimaryPort” and **“ServerPrimaryPortSSL”** - the DirX Access server ports.

“oauth2AuthServer”

the connectivity URL to DirX Identity REST Services for data and connectivity URL to OAuth2 Identity Providers. You must configure the following parameters:

DirX Identity REST parameters:

“protocol”

the protocol, either http or https. If the Business User Interface is accessed through HTTPS, you can only use HTTPS to access the DirX Identity REST Services.

“host”

the name of the server on which the DirX Identity REST Services is running. If you use HTTPS, you must use the same name as one of the subject alternative names of the server certificate.

“port”

the port on which the DirX Identity REST Services is accessible via HTTP or HTTPS. Configure **80** for the standard HTTP port. Configure **443** for the HTTPS standard port.

“domain”

the base path to the DirX Identity REST Services, which also configures the domain for which the Business User Interface is used.

The domain field contains the DirX Identity REST services context path and the DirX Identity domain; for example: /DirXIdentityRestService-My-Company. See the chapter “DirX Identity

REST Services" in the DirX Identity Integration Framework Guide for more information.

OAuth2 parameters:

"clientId"

the application ID used in OAuth2 Identity Provider configuration. The Business User Interface application must be registered to Identity Provider with this ID.

"issuer"

the URL to the OAuth2 provider. Here are issuer examples for different OAuth2 providers:

For DirX Access:
https://my-server.my-company.example:11115/auth

For RedHat Keycloak:
http://localhost:8180/auth/realms/my-company

"timeoutFactor"

optional - the refresh token acquisition rate. For example, with a value of **0.75**, a new access token will be automatically requested when 75% of the access token validity time has passed.

Token expiration time is configured on Identity Provider settings.

"security" - "sessionTimeout"

the number of seconds after which a session is invalidated after the idle state has been detected.

"security" - "sessionIdle"

the number of seconds after which the idle state is indicated.

"security" - "authentication"

the login methods to be displayed in the login dialog:

["BASIC"]

shows only the username/password login method using the REST interface configured in **"basicAuthServer"**.

["X509"]

shows only the PKI login method using the REST interface configured in **"x509AuthServer"**.

["KERBEROS"]

shows only the Windows Authentication login method using the REST interface configured in **"kerberosAuthServer"**.

[“DXAPEP”]

shows only the DirX Access Policy Enforcement Point login method.

[“OAUTH2”]

shows only the OAuth2 OpenID login method using configuration from “**oauth2AuthServer”.**

[“BASIC”, “X509”, “KERBEROS”]

shows the username/password login method with the ability to switch to the PKI login method or Windows Authentication login method.

[“X509”, “BASIC”]

shows the PKI login method with the ability to switch to the username/password login method or Windows Authentication login method.

[“KERBEROS”, “BASIC”, “X509”]

shows the Windows Authentication login method with the ability to switch to the username/password login method or PKI login.

[“BASIC”, “OAUTH2”, “X509”, “KERBEROS”, “DXAPEP”]

shows all supported authentication methods with default selection to basic authentication method.

[]

disables any login method (useful for maintenance mode).

“security” - “authenticationAttributeNames”

shows the attribute names used for authentication. These attributes are used to display information in the home page (emails, name) and attributes used to configure the application (**preferredLanguage**, **dxrpreferences**).

If the feature “enforce change password on next login” is used, add the following line to the security entry: **“passwordResetAttributeName”: “dxrPwdReset”**.

We strongly recommend that you do not change this attribute names list. Incorrect or missing attribute names may interfere with the authentication process.

These settings must correspond to the settings made in the Tomcat server and the DirX Identity REST Services.

The Tomcat server is responsible for accepting PKI client authentications.

The DirX Identity REST Services are responsible for mapping the PKI client certificate to a valid DirX Identity user and/or for mapping the username and password to a valid DirX Identity user.

The Tomcat server must be configured for this function in the file:
*tomcat_install_path***/conf/server.xml**

The DirX Identity REST Services must be configured for this function in the file: *install_path*/**restServices/DirXIdentityRestServices/DirXIdentityRestService-*domain*/WEB-INF/security.properties**

Example 1. Configuration

Your server is named **myserver.mycompany.net** and your DirX domain is named **My-Company**.

Tomcat configuration 1: Tomcat accepts username and password requests over HTTPS on port **443**.

Tomcat configuration snippet:

```
<Connector port="443"
protocol="org.apache.coyote.http11.Http11NioProtocol"
maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS"
keystoreFile="install_path\ssl\identity-keystore"
keystorePass="changeme" />
```

The **keystoreFile** must contain a valid server certificate for the Web service and the server name **myserver.mycompany.net**, which must be trusted by the browser in use.

Tomcat configuration 2: Tomcat accepts PKI client authorization requests over HTTPS on port **8443**.

Tomcat configuration snippet:

```
<Connector port="8443"
protocol="org.apache.coyote.http11.Http11NioProtocol"
maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
clientAuth="true" sslProtocol="TLS"
truststoreFile=" install_path\ssl\identity-truststore"
truststorePass="changeme"
keystoreFile="install_path\ssl\identity-keystore"
keystorePass="changeme" />
```

The **keystoreFile** must contain a valid server certificate for the Web service and the server name **myserver.mycompany.net**, which must be trusted by the browser in use.

The **truststoreFile** must contain the root certification authority (CA) certificate and perhaps the intermediate CA chain, which has signed the client certificates to be used to access the service.

Tomcat configuration 1 and configuration 2 can be used in parallel inside one Tomcat instance.

Your Business User Interface configuration for basic and x509 authentication methods snippet should look like this:

```
"basicAuthServer": {
        "protocol": "https",
        "host": "myserver.mycompany.net",
        "port": "443",
        "domain": "/DirXIdentityRestService-My-Company"
    },
    "x509AuthServer":{
        "protocol": "https",
        "host": "myserver.mycompany.net",
        "port": "8443",
        "domain": "/DirXIdentityRestService-My-Company"
    },
    "security": {
        "authentication": ["X509","BASIC"],…
```

and you can access the Business User Interface with this URL in your browser: https://myserver.mycompany.net/BusinessUserInterface-My-Company/

DirX Identity REST Services example for PKI login:

If using PKI client authentication, the DirX Identity REST Services are configured by default to map the first appearance of a CN=-part inside the client certificates subject field to the user's **cn** field inside the DirX Identity LDAP directory.

DirX Identity REST Services configuration snippet:

```
auth.userFilter     = (cn={0})
auth.valueSelectRDN = cn:1
```

If you want to map the SERIALNUMBER part of a client certificates subject field to the **employeeNumber** of the DirX Identity LDAP user entry, you can use the following snippet:

```
auth.userFilter     = (employeeNumber={0})
auth.valueSelectRDN = SERIALNUMBER:1
```

1.4.2. Adding a New Language

Adding a new language consists of two steps:

- Configuring the new language in the configuration file
- Preparing a new language file

1.4.2.1. Configuring the New Language in the Configuration File

You need to add the language to the **config.json** file located in the Business User Interface installation folder *install_path***/web/BusinessUserInterface-***domain***/assets/config/**.

In the **"languages"** section, you'll find the two preconfigured languages English and German ("Deutsch (DE)"). Let's add a new language "Français (FR)". To do this, add a new section between the English and the German section:

```
{
    "displayName": " Français (FR)",
    "code": "fr"
},
```

The language definitions must be separated from each other by a comma.

The **code** keyword defines the name of the language file you need to prepare. In our example, the value is **fr**.

You can also define different language flavors for the same base language. For example, you can add the language flavor "English (GB)" with the code **en_GB**.

1.4.2.2. Preparing the Language File

The language filename must be in the format *code**.json* and must be located in *install_path***/web/BusinessUserInterface-***domain***/assets/i18n/**.

For our French example, the filename is **fr.json**, for the British example, it is **en_GB.json**.

The best way to prepare a new language file is to take the **en.json** file from the deployment as a template and then translate all of the values (not the parameter names!) into the new language. The values are always the part behind the colon (**:**). In some cases, the values contain dynamic parts, which can be recognized as \{\{...}}. The dynamic part is not to be translated if used inside a value.

1.4.3. Exchanging Images and Graphics

You can replace some of the graphics of the Business User Interface according to your requirements. The base path of the graphics is:
*install_path***/web/BusinessUserInterface-***domain***/assets/**.

You can change the following graphics and images (you must keep the filename and just

replace the default image or graphic):

img/company-logo-contrast.svg
: vector image (svg); blue on transparent;aspect ratio 1.41/1; company logo for header information.

img/company-logo.svg
: vector image (svg); white on transparent; aspect ratio 1.41/1; company logo in burger menu and for busy animation at startup.

img/company-logo.png
: png file; white on blue; 211*149 pixels; company logo for login.

img/company-background.jpg
: jpg file; colored and non-transparent; 1920*1244 pixels (recommended); background image for login.

Other graphics/images include:

img/anonymous.svg
: vector image (svg); white on transparent; aspect ratio 1/1.

Img/company-logo.svg
: vector image (svg); white on transparent; aspect ratio 1.41/1.

You can replace the icon *install_path***/web/BusinessUserInterface**-*domain***/**assets/icon/**favicon.ico** with one of your company's icons.

1.4.3.1. Customizing Tables

These settings are available in the **tables.json** file located in the Business User Interface installation folder:
*install_path***/web/BusinessUserInterface-***domain***/assets/config/**.

This section contains three main subsections: "my" ,"myTeam" and "userManagement". Each subsection describes the table content for the "my" section of the application (for example, "My Tasks", "My Access Rights"), and the "myTeam" and "userManagement" sections (for example, "Manage access rights", "Manage requests".).

Each subsection contains entries for each table that are available in that page. The general format contains entries for the default columns provided by the Business User Interface, optional custom columns entries, and a list of table attributes.

Table attributes are described by the following parameters:

rowsLimits
: a number that specifies the number of visible rows per table page. The default value is **"5"**.

columnMode

a constant that specifies the column mode width calculation. Available options are **force** – distributes proportionally, **flex** – uses flex grow, and **standard** – distributes based on **width** parameter. The default value is **“standard”**.

Default columns are described by the following parameters:

id (mandatory)

a string that specifies the ID of the column.

visible (mandatory)

whether (**true**) or not (**false**) the custom column is visible.

width (optional)

the width of the table column, can be entered with any CSS type values. For example, **rem**, **px**, or **%**.

sortable (optional)

whether (**true**) or not (**false**) the column is sortable.

Custom columns are optional and are described by the following parameters:

type (mandatory)

a string whose value is always **custom**.

id (mandatory)

a string that specifies the ID of the column.

attributeName (mandatory)

a string that specifies the name of the attribute assigned to the column. This field is a reference for implementation in the Business User Interface hook. It should provide sufficient information for hook to read the value for the custom column from the model (for example, “attribute”: “assignment.mode”). See the section “Using Hooks to Extend the Business User Interface” for details.

visible (mandatory)

whether (**true**) or not (**false**) the custom column is visible.

width (optional)

the width of the table column, can be entered with any CSS type values. For example, **rem**, **px**, or **%**.

sortable (optional)

whether (**true**) or not (**false**) the column is sortable.

Here is an example:

```
"tables": {
```

```
    “my”: {
"tasks": {
“tasks”: {
    “rowLimit”: 5,
    “columnMode”: “standard”,
“columns”:[
                    {
                        "id": "avatar",
                    "visible": true,
                        “width”: 75,
                        “sortable”: true
    },{
        "type": "custom",
        "id": "description",
        "attributeName": "description",
        "visible": false,
            "width": "30%"
    }]
}
```

Default column values for "tasks" are:

"tasks": name, dueDate, operation, endDate, roleParameters, sod, risk, action, summary

Default column values for "accessRights" are:

"roles": name, startDate, endDate, roleParameters, mode, state, action, summary

"permissions": name, startDate, endDate, mode, state, action, summary

“groups": name, startDate, endDate, mode, state, action, summary

Default column values for "privilegeRequest" are:

"roles": name, startDate, endDate, roleParameters, action, summary

"permissions": name, startDate, endDate, action, summary

"groups": name, startDate, endDate, action, summary

Default column values for "requests" are:

“roles”: reqType, display, initiator, subject, nextApprovalDate, currentParticipants, operation, startDate, endDate, roleParameters, hasSODViolations, actions, summary

"permissions": reqType, display, initiator, subject, nextApprovalDate, currentParticipants, operation, startDate, endDate, hasSODViolations, actions, summary

"groups": reqType, display, initiator, subject, **nextApprovalDate, currentParticipants, operation**, **startDate**, **endDate**, **hasSODViolations, actions, summary**

"profileChanges": reqType, display, initiator, subject, **nextApprovalDate, currentParticipants, operation**, **startDate**, **endDate**, **roleParameters, hasSODViolations, actions, summary**

Default column values for "myTeam" are:

“users”: avatar, name, **emails**, **phones**, **riskLevel**, **state**, **userHome**

Default column values for “userManagement” are:

“users”: avater, name, emails, phones, risk, state, action, username

Default column values for “myDelegations” are:

“delegations”: name, operation, assignTo, startDate, endDate, futherDelegations, action

“assignedDelegations”: name, operation, assignFrom, startDate, endDate, futherDelegations, action

Default column values for “myCampaigns” are:

“campaigns”: **type, name, startDate**, **dueDate**, **owner**, **ownerMail, ownerTelephoneNumber**, **numberOfCerifications, gotoCertifications**

Defalut column values for “myCampaign” are:

“**users”: username, dueDate, completed, gotoTasks**

“**privileges”: privilegeName, folder, dueDate, completed, gotoTasks**

Default column values for “certify” are:

“users”: approveState, rejectState, reason, name, department, startDate, endDate, roleParameters, sod, summary

“roles”: approveState, rejectState, reason, name, startDate, endDate, roleParameters, sod, summary

“permissions”: approveState, rejectState, reason, name, startDate, endDate, sod, summary,

“groups”: approveState, rejectState, reason, name, targetSystem, startDate, endDate, sod, summary,

1.4.4. Adding a New Widget

These settings are available in the **config.json** file in the section “customWidgets” located in the Business User Interface installation folder: *install_path***/web/BusinessUserInterface-***domain***/assets/config/**.

This section can be used to extend the default home page with additional widgets. These

widgets support only external links to other websites.

Here is an example:

```
{
"id": "home-my-atos-id",
    "title": "My Atos",
    "label1": "My Atos",
    "label2": "external link",
    "customIcon": "./assets/img/home-atos.png",
    "link": "https://atos.net"
}
```

1.4.5. Enabling and Disabling Different Features

These settings are available in the **config.json** file located in the Business User Interface installation folder:
install_path/**web/BusinessUserInterface-***domain*/**assets/config/**.

1.4.5.1. Using the Localization Feature

This feature allows the Business User Interface application to adapt displayed content (currency, date, time) to different languages, regional peculiarities, and technical requirements of a target locale.

The Business User Interface configuration inside the **“ui”** section: “**localeStrategy**” sets the order of locale strategy usage:

“ui” - “localeStrategy” - “profileAttribute”
: the Business User Interface application uses the LDAP attribute name set in this section. If DirX Identity does not use this attribute, set this field to **“null”** or remove this key.

“ui” - “localeStrategy” - “browser”
: the Business User Interface application uses the internet browser settings for displaying localized content. To disable this strategy, set the value to false or remove this key.

“ui’ – localeStrategy” – “fallback”
: if none of the strategies listed above are used or enabled, the Business User Interface uses this key. This key is the same for all Business User Interface users.

These localization settings do not affect internationalization settings.

1.4.5.2. Disabling the Risk Feature

If you have not enabled the risk governance feature in your DirX Identity domain, you must disable it in the Business User Interface.

To disable this feature, set the parameter **“showRiskLevel”** Inside the **“ui”** - section to “ **false**”. The default value is “**true**”.

1.4.5.3. Enabling/Disabling the Display of Privileges

You can choose which kind of privileges (roles, permissions, groups) are displayed.

To suppress the display of different kinds of privileges, set the parameters **“showRoles”**, **“showPermissions”, “showGroups”** and/or **“showProfileChanges”** Inside the **“ui”** - section to “**false**”. The default value for all three is “**true**”.

1.4.5.4. Enabling/Disabling the Profile Panels

You can choose to display the name initials instead of the profile photo. To enable this feature, set the parameter **“showAlwaysInitials”** Inside the **“ui”** - section to “**true**”. The default value is “**false**”.

You can also choose to prevent the display of the name initials. To enable this feature, set the parameter **“showInitials”** Inside the **“ui”** - section to “**false**”. The default value is “**true**”.

The “My Team” and “User Management” pages show initials in the users list.

You can also select which panels should be displayed in Profile pages:

“show<My|User>ProfileAvatar’
display or hide the user avatar panel. The default value is **“true”**.

“show<My|User>ProfileOperational’
display or hide the user operational attributes panel. The default value is **“true”**.

“show<My|User>ProfileCertificates”
display or hide the user certificates panel. The default value is **“true”**.

“showClosedRequests”
display or hide closed authenticated user requests. The default value is **“true”**.

“showWorkflowRequestsAsInitiator”
display or hide requests where the authenticated user acts as the initiator. The default value in **“true”**.

“showTicketRequestsAsInitiator”
display or hide tickets where the authenticated user acts as the initiator. The default value is **“true”**.

“showPersonas”
display or hide users of type persona. The default value is **“true”**.

“showUserFacets”
display or hide users of type user facet. The default value is **“true”**.

“showFunctionalUsers”

display or hide users of type functional user. The default value is **“true”**.

1.4.5.5. Enabling/Disabling Home Page Counts

To improve home page performance, you can disable home page counts: the values displayed on every widget, such as the number of tasks, the number of access rights, and so on. The list of home page counts is available in the **“home_counts_ids.txt”** file in the folder *install_path***/web/BusinessUserInterface**-*domain***/assets/config/**.

1.4.5.6. Defining Color Schemes in the “themes” Subsection

You can create new color schemes for the Business User Interface. To use these new schemes, you must register them in the configuration by providing the following parameters:

name

the name of the theme (string).

file

the name of the CSS file that contains the theme (string). This file must be placed in the folder *install_path**/web/BusinessUserInterface-***domain***/assets/config/styles/*.

colors

the colors to be displayed in the Business User Interface Settings page as the representation of the theme (the primary colors).

1.4.6. Configuring Visible Widgets

The visible widgets available in the Business User Interface can be configured with the following parameters:

accessPolicyId

the access policy ID assigned to the current widget. Available access policy IDs are defined in the **policy.json** file and are prefixed with the **home-****string* **(for example, *home-my-profile-id**).

displayName

the translation ID for the current widget. Available translation IDs are defined in the file *code**.json* in the translation folder and are prefixed with **HOME.****string* **(for example, *HOME.MY_PROFILE_TITLE**).

i18n

the translation ID for the current widget. Available translation IDs are defined in the file *code**.json* in the translation folder and are prefixed with **HOME.****string* **(for example, *HOME.MY_PROFILE_TITLE**).

visible

whether (**true**) or not (**false**) the widget is visible. This value can be overwritten by DirX

Identity access policies.

section

the sections to which the widget belongs. Available values are **“my”**, **“myTeam”**, and **“userManagement”**. These values are only for information and cannot be changed.

1.4.7. Configuring Access Policies in the Business User Interface

The access policies configuration file **policy.json** for the Business User Interface is located in the folder *install_path*/**web/BusinessUserInterface**-*domain*/**assets/config/** and is applied to all users.

Access policies actions are executed on the Tomcat server. In the Business User Interface, they exist to present to the user when a field can be visible, when it can be read and when it can be modified. This file contains information from DirX Identity Manager and should be updated when changes are made to access policies in DirX Identity.

JSON file content is case sensitive. Make sure that you follow the correct JSON syntax and case format for keys and values.

The file contains one entry: **“access-policies-map”**. This map contains entries for each menu (a Business User Interface widget) and each entry (a Business User Interface page or panel).

Each entry from **“access-policies-map”** contains the following fields:

disabled (mandatory)

whether (**false**) or not (**true**) access policies are applied in the Business User Interface. If access policies are disabled in DirX Identity, this parameter ***must*** be set to **true**.

***component_id* (mandatory)**

a string that specifies the ID of a Business User Interface component, such as a page or an action button. For example, "my-profile-edit-id": \{ ... } - specifies the ID of the “my-profile” page for the edit button.

The “access-policies-map” contains all of the IDs from the Business User Interface application. An access policy ID is not the same as the HTML component ID.

type (mandatory)

a string that specifies the type of access policy to be applied to the component. Supported values are **menus**, **entry** and **<empty>** (**<empty>** is JSON syntax for an empty string “ ”).

menus

the component is linked to an access policy item in the “menus” section of the DirX Identity Manager GUI’s “Policy” section.

entry

the component is linked to an access policy item in the “entry” section of DirX Identity Manager GUI’s “Policy” section.

<empty>

the component is not linked to a DirX Identity Manager access policy item.

menu-key (mandatory)

a string that specifies the access policy’s context of execution: a menu or an entry. For example, **"menu-key": "my-profile"** specifies that the policy is assigned to the “my-profile” panel entry context. Possible values are **“my-profile”** and **“user-profile”**.

For possible values for type **“menus”**, see the “ResourceTypes” entry from the **resource.json** file available in the REST Services configuration folder.

menu-item (mandatory) - a string that specifies the operation to be executed.

The possible values are “**read”** and “**modify”**. For type **“menus”**, see the DirX Identity Manager “Policies” section for supported operations.

display (mandatory) - whether (**true**) or not (**false**) the entry is visible. This field can be used to hide the component even if the access policy allows the component to be displayed.

1.4.8. Using Dynamic Forms to Customize User Profiles

You can use the dynamic form framework Angular Formly (https://github.com/ngx-formly/ngx-formly/) to customize the “My Profile” and “Manage User Profile” pages. This framework allows you to add input fields in a form at runtime and additional entries to the user profile presentation. This section describes how to extend the Business User Interface with additional user attributes.

Configuration and description files for dynamic forms are available in *install_path***/web/BusinessUserInterface**-*domain***/assets/config/forms/my-profile.json** for the “My Profile” page and **user-profile.json** for “Manage User Profile”. By default, these files are identical and can be changed and customized to your requirements.

The content of a dynamic form is described in a JSON file. The JSON file contains two sections: “attributeNames” and “fields”. Fields of the form are described in the “fields” section and referenced in the “attributeNames” section. Each field contains a list of mandatory and optional Formly and BUI attributes. BUI attributes are identified by the prefix **bui** and are described in this document. For information about the Formly attributes, see the Formly product documentation available on the Formly website.

JSON file content is case sensitive; please make sure that you follow correct syntax and case format for keys and values.

1.4.8.1. The “attributeNames” Section

The “attributeNames” section contains one entry that consists of a simple array of SCIM and/or LDAP attribute names. To extend a user profile with an attribute, add the attribute

name to this section and then describe it in the “fields” section. You can also supply an attribute name in this section that does not participate in the profile visual presentation; for example, the **dxrUID** attribute. This attribute is part of the profile data model but does not have a visual representation in the user profile page, so it does not need a corresponding entry in the “fields” section.

Here is an example of the “attributeNames” section:

```
"attributeNames": [ "emails", "c", "meta", "schemas", "name",
"emails", "phonenumbers"]
```

1.4.8.2. The “fields” Section

The “fields” section describes how the attributes are represented in the page. Each field must have a corresponding entry in the “attributeNames” section.

A field contains following entries:

key (mandatory)
: a string that represents the name of the SCIM/LDAP attribute assigned to the dynamic field. This is a Formly attribute. For example, "key": "manager" specifies that this field is assigned to the “manager” SCIM attribute.

In some cases, the SCIM/LDAP attribute can contain multiple subattributes. In this case, use the character “**#**” to specify a particular subattribute. For example, "key": "emails#work" assigns this field to the “emails” SCIM attribute, and from this attribute only the “work” attribute is used.

type (mandatory)
: a string that specifies the HTML component type: **select**, **textarea**, and so on; for example, "type": "select". See the Formly website for a complete list of the supported HTML component types.

The Business User Interface provides a custom type **“buiselect”** that allows you to perform searches to LDAP and retrieve complex attributes. Mandatory entries for the **buiselect** type in the **templateOptions** subsection are **bui_displayAttributes**, **bui_displayKey**, **bui_search_action** and **bui_searchAttributes**.

templateOptions (mandatory)
: a subsection required for each field (Angular Formly configuration syntax).

bui_i18n (mandatory)
: a string that specifies the corresponding key from the translation file available in the *install_path*/web/BusinessUserInterface*-*domain*/assets/i18n/* folder. For example, the field "bui_i18n": "PROFILE.MANAGER_LABEL" has the translation from the “PROFILE.MANAGER_LABEL" key from the translation file.

bui_displayAttributes (mandatory for “buiselect” type)

a subsection that describes which attributes will be used to display search results. It contains two entries: **title** and **subtitle**. Here is an example:

```
"bui_displayAttributes": {
    "title": "name.formatted",
    "subtitle": "ou"
},
```

In this example, for a “buiselect” type component and the “manager” key, the search results are visually represented in the dialog by “name.formatted” as the main title and “ou” as the subtitle.

bui_displayKey (mandatory for “buiselect” type)

a string that describes which entry for the model is used for display by the component.

bui_searchAction (mandatory for “buiselect” type)

a string that describes the type of search to be executed by the REST Services. For a complete list of the supported search actions, consult the file **resources.json** in the REST Services location. For example, `"bui_searchAction": "managers"` directs the REST Service to execute a search for “managers” and return the results.

bui_searchAttributes (mandatory for “buiselect” type)

a subsection that describes the expected attributes in a result entry from a search action. For example, `"bui_searchAttributes": [ "id", "name", "ou" ]` indicates that the result should contain the attributes **id**, **name**, and **ou**.

bui_valueKey (mandatory for “buiselect” type)

a string that describes the attribute used which contains the value of the entry. For example, the bui_displayKey represents the visual representation of the entry (Olivier Hungs) and bui_valueKey represents the data (00014281).

bui_options (mandatory for “select” type)

a string that specifies a reference to a list of JSON subsections in the **config.json** file with “key” as the handler and “name” as the display value. Here is an example from the “isoLanguages” entry in **config.json**:

```
"isoLanguages" : {
"af": {
    "name": "Afrikaans",
    "nativeName": "Afrikaans"
    },
     …
}
```

bui_masterAttribute (mandatory for indirect attributes)

a string that specifies an LDAP attribute name to be used to store the date at which the profile is saved. For example, the SCIM attribute “ou” is an indirect attribute; the information is stored in the “dxrOULink” LDAP attribute. When the field “ou” is saved, the value which is described in “bui_valueKey” is stored to the LDAP attribute described in “bui_masterAttribute”.

bui_readonly (optional)

whether (**true**) or not (**false**) the attribute is read only. You can use this field to display an attribute that is not to be modified.

bui_separator (optional)

a string to be used instead of the “new line” control character (<ENTER>) for the “textarea” field type.

1.4.9. Using Hooks to Extend the Business User Interface

Internet Explorer 11 requires an old JavaScript syntax. If IE11 is used, make sure you use the proper syntax for JavaScript code used in hooks. Some features and syntax format may be supported with polyfills in BUI. Test your hooks code with IE11 (Edge engine) to validate the code from hooks.

The Business User Interface application provides custom extensions to the following tables:

Page	Hook Entry
My Tasks	tasksComponentHook
My Access Rights	accessRightsComponentHook
Privilege Request	privilegeRequestComponentHook
My Requests	requestsComponentHook
My Team	myTeamComponentHook
Manage Access Rights	accessRightsComponentHook
Manage Requests	requestsComponentHook
My Delegations	myDelegationsComponentHook
My Campaigns	myCampaignsComponentHook
My Campaign	myCampaignComponentHook
Certify	certifyComponentHook
User Management – Privilege Request	userMgmtPrivilegeRequestComponentHook
Profile Service	profileServiceHook
Privileges Service	privilegesServiceHook
Certificates Service	certificatesServiceHook
My Team Service	myTeamServiceHook

Page	Hook Entry
Certification Campaigns Service	certificationCampaigsnServiceHook
Request Service	requestServiceHook
User Management Service	userManagementServiceHook

These extensions are available in the file *install_path*/**web/BusinessUserInterface-***domain*/**extern/hook.js**.

JSON file content is case-sensitive. Make sure you follow correct syntax and case format for functions.

Each extension provides the following functions:

- **search: function(entry, text)** - use this function to extend the search capabilities for a table. If the table contains custom columns, you can use this function to extend the search for custom entries. Parameters are:

  entry
  : specifies the data model to search. Use this parameter to search for specific content and/or the text provided by the **text** parameter.

  text
  : specifies the text to search.

  This function returns true if the search in the model has results, false if the search has no results and undefined if the search is not relevant for current entry.

- **sort: function(field, a, b, direction)** - use this function to sort a column from a table by comparing two entries for the model (**a** and **b** parameters). Parameters are:

  field
  : a string that specifies the column name

  a
  : specifies the first entry

  b
  : specifies the second entry

- **customContent: function(entry, customColumn)** - use this function to format custom attributes. The expected result from this function is a string that may contain HTML tags.

The customColumn parameter contains attributes defined in the “tables” section of the **config.json** file. From this entry, you can use the id and attribute entries to retrieve information from the entry data model. For example, the “attribute” entry may contain a “description” value. You can use this value to extract the value from the model and return a formatted string. Here is an example:

In **config.json**, “tables” section:

```
{
"type": "custom",
    "id": "description",
    "title": "Desc",
    "tooltip": "Description",
    "attribute": "description",
    "visible": true
}
```

In **hook.js**, “tasksComponentHook” section:

```
customContent: function (entry, customColumn) {
var value = entry[customColumn['attribute'];
value = '<b>'+value+'</b>' // format text to bold style
    return value;
}
```

1.4.10. Customizing Attributes from Requests

Note: Be careful when changing the values in these sections.

In the section “**requests**”, **“delegations”, “myTeam”, “userManagement”, “search”, “certificationCampaigns”, “accessRights”, “privilegeRequests”, “domainObjects”** in **config.json**:

> **“requests”** - **“searchParticipantAction”** - *string* – shows the action name used for the “change participants” feature for “requests” pages. This entry must contain an action defined in the REST service configuration file (**resources.json**). By default, this value is set to **“managers”**.
>
> *section* - **“attributeNames”** - *string* – contains a list of attribute names (SCIM or LDAP) that can be added to a BUI request to REST services. The REST service can reject these attributes so that they are not available in the REST response. These attributes can be used in user hooks implementations combined with custom columns in data tables.

1.4.11. Debugging Customizations

The Business User Interface does not write information in DirX Identity log files. You can find information about access requests inside Apache Tomcat in the access logs (*domain-name*access_log.**_timestamp**.txt*).

To get more information about the triggered DirX Identity REST Service requests, you can use the debugging feature of your browser (Key F12) in your Business User Interface

session.

1.4.11.1. From Chrome

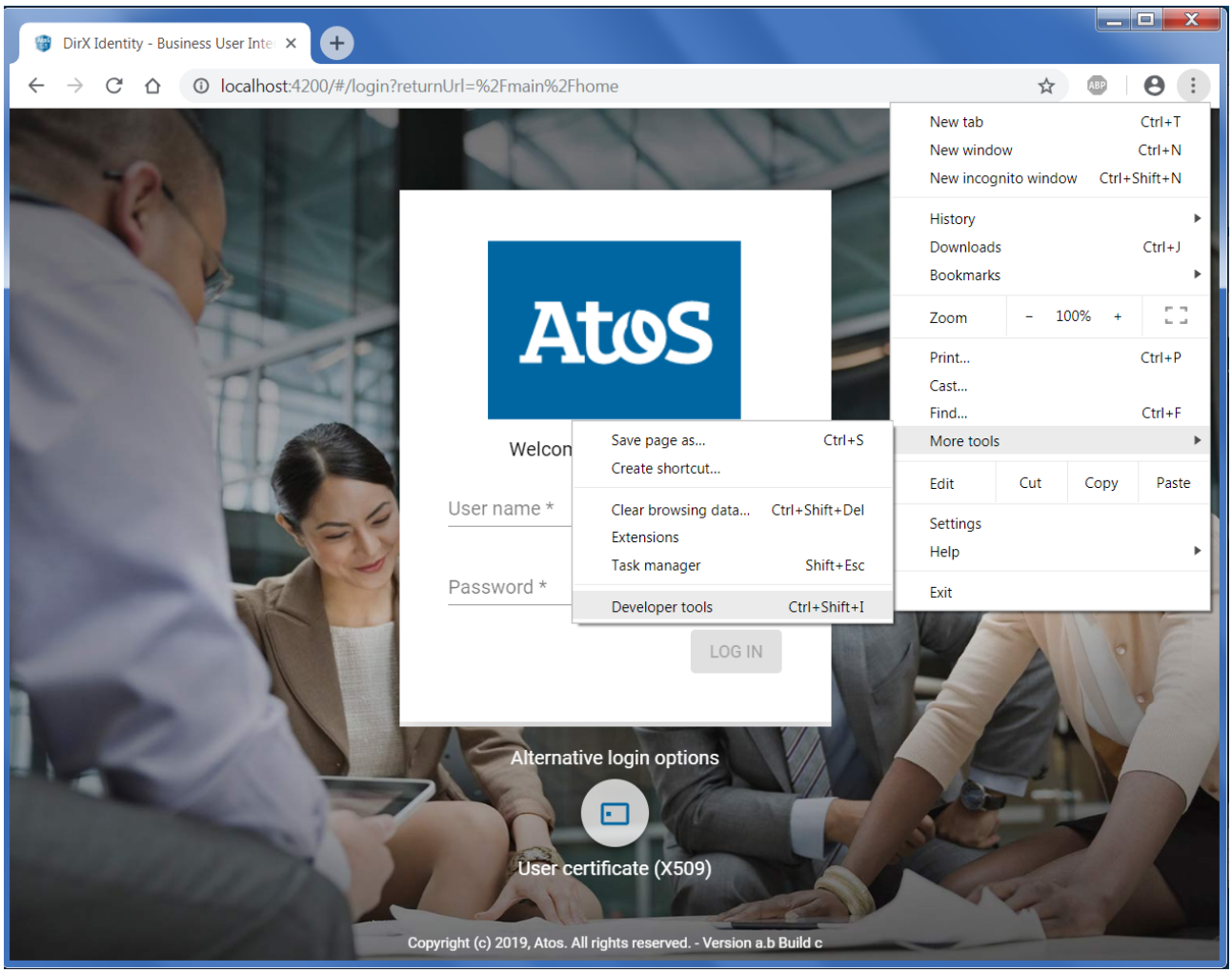

Figure 5. Chrome Developer Tools Menu

From the Chrome menu, select **“More tools”** > **“Developer tools”** or press the shortcut keys combination “Ctrl+Shift+I” or press the F12 key.

A new page is displayed either attached to the current page or as an individual page:

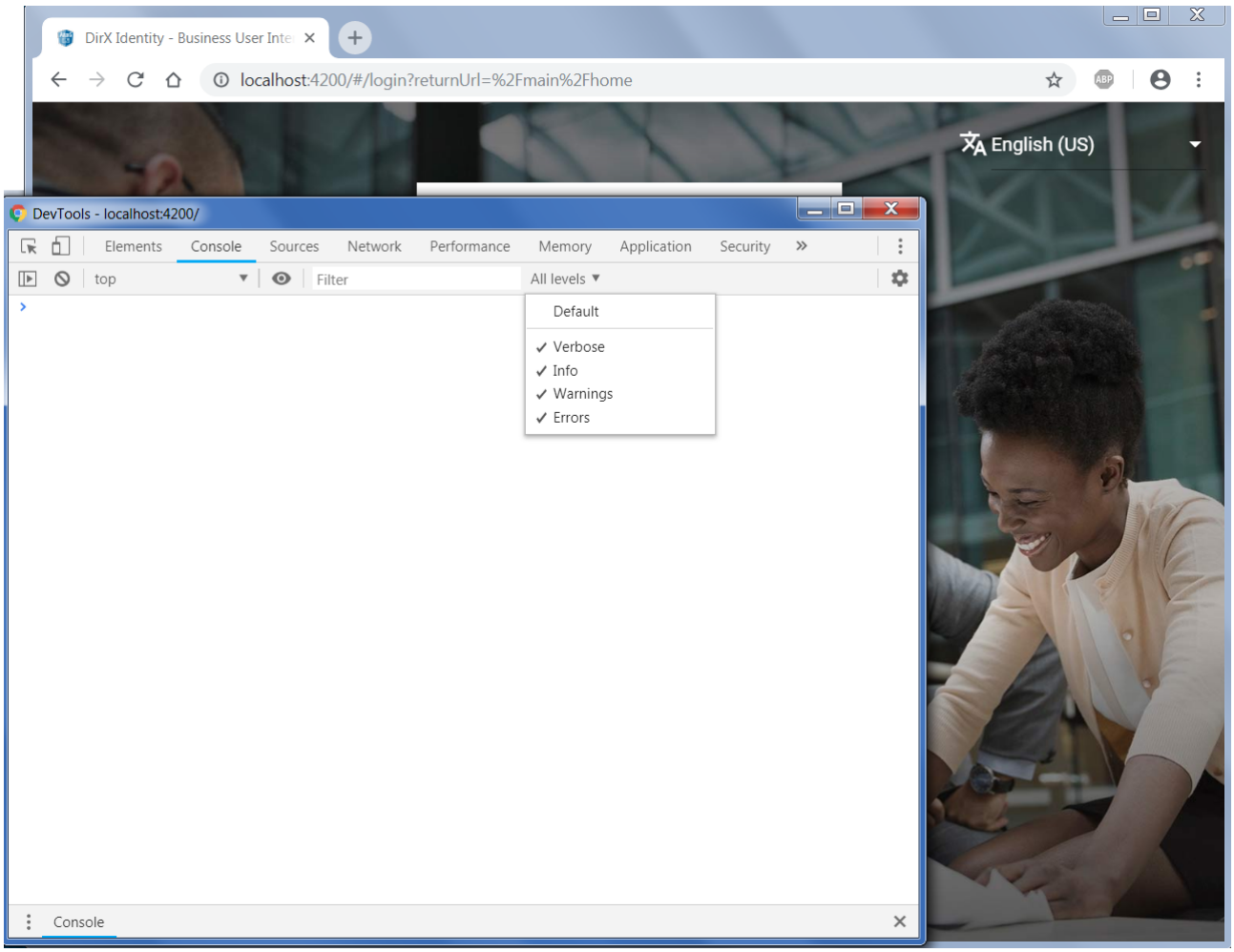

Figure 6. Chrome Developer Tools Window

Select the “Console” tab. From the toolbar entry “All level”, select an appropriate debug level (**Verbose** - high detail debug information, **Errors** - displays only error information).

Clear the console (Ctrl+L) and execute actions in the application window until the error is reproduced. Return to the “Developer tools” window and copy and paste the content into a text file. This content contains the execution logs from the Business User Interface application.

1.4.11.2. From Firefox

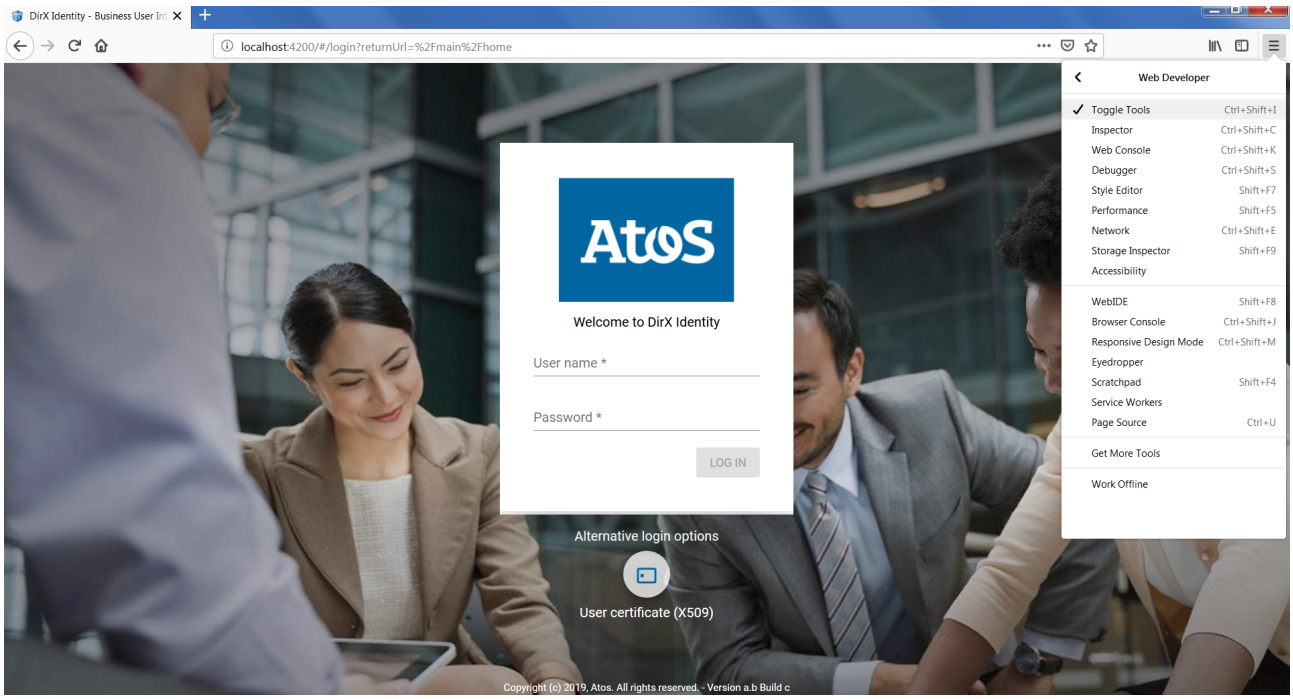

Figure 7. Firefox Web Developer Window

From the Firefox menu, select **“Web Developer” – “Web Console”** or press shortcut keys combination “Ctrl+Shift+K” or press key F12 and select the Console tab.

A new page is displayed either attached to the current page or as an individual page.

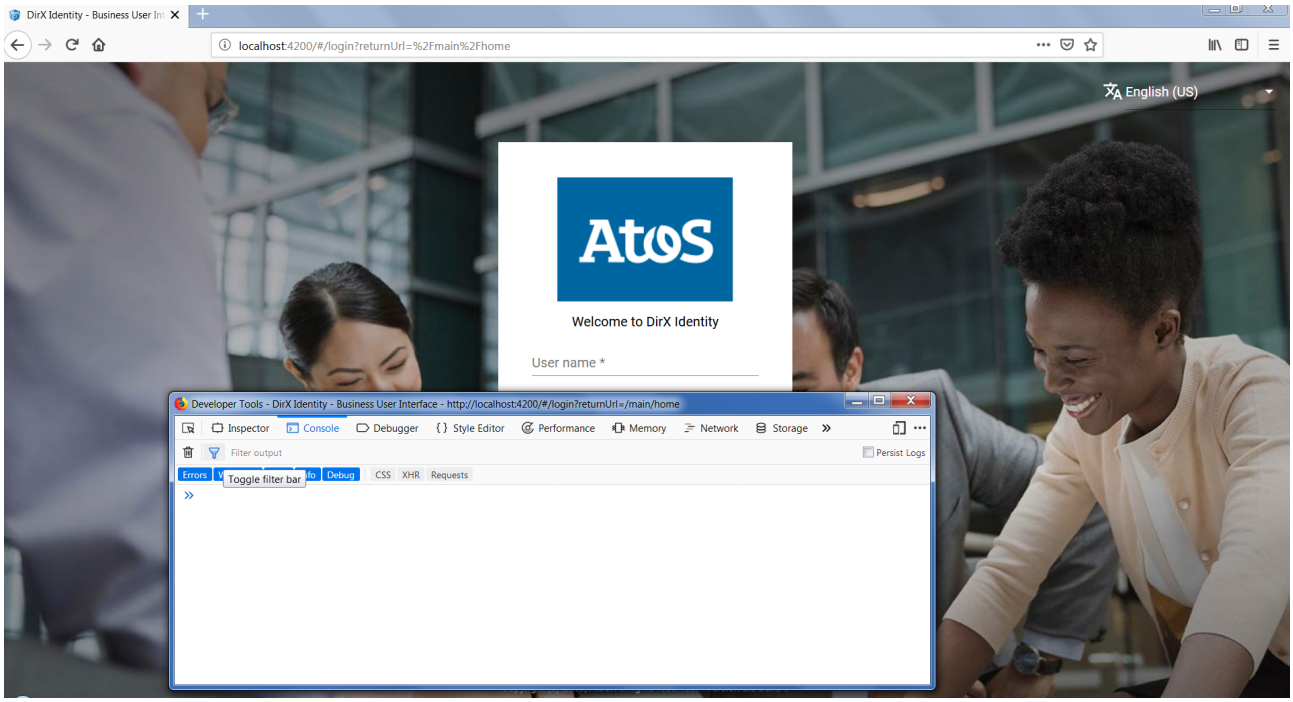

Figure 8. Firefox Web Developer Window

Select the tab “Console”. From the toolbar, select an appropriate debug level (check or uncheck options: Errors, Warnings, Logs, Info, Debug).

Clear the Web console output and execute actions in the application window until the error is reproduced. Return to the “Developer tools” window and copy and paste the content into

a text file. This content contains the execution logs from the Business User Interface application.

1.4.11.3. From Internet Explorer

Figure 9. Internet Explorer Developer Tools Menu

From the Internet Explorer menu, select **“F12 Developer Tools”** or press the F12 key and select the Console tab.

A new page is displayed either attached to the current page or as an individual page.

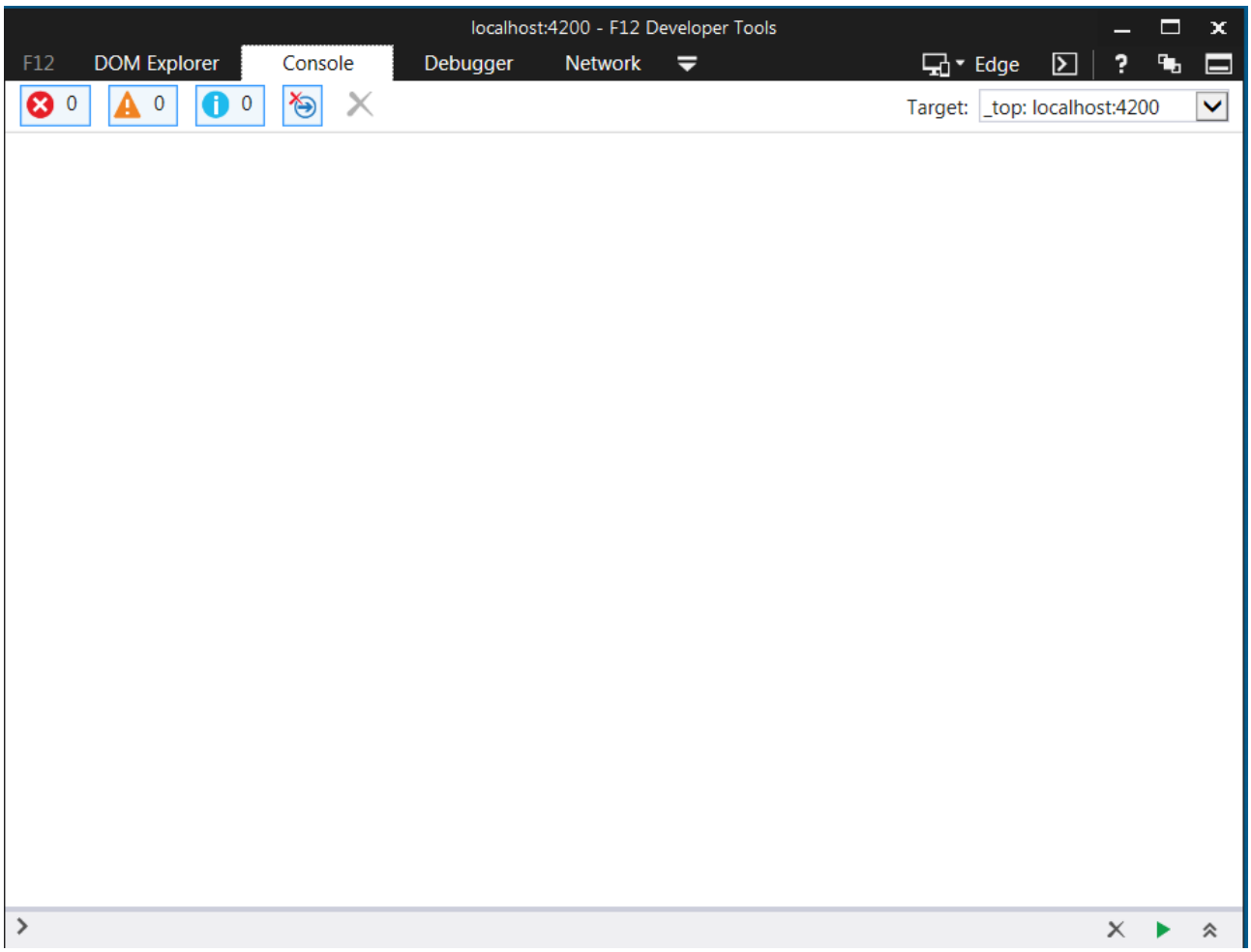

Figure 10. Internet Explorer Developer Tools Window

Select the “Console” tab and then select an appropriate debug level (check or uncheck icons).

Clear the Web console output and execute actions in the application window until the error is reproduced. Return to the “F12 Developer tools” window and copy and paste the content into a text file. This content contains the execution logs from Business User Interface application.

DirX Identity REST Services logging is described in the *DirX Identity Integration Framework Guide*. See the section “Configuring Application Logging” in the section “Configuring the DirX Identity REST Services” in the chapter “DirX Identity REST Services”.

1.4.12. Customizing Login Information

Administrators have the option of prompting users with a system-relevant message such as scheduled maintenance. For example:

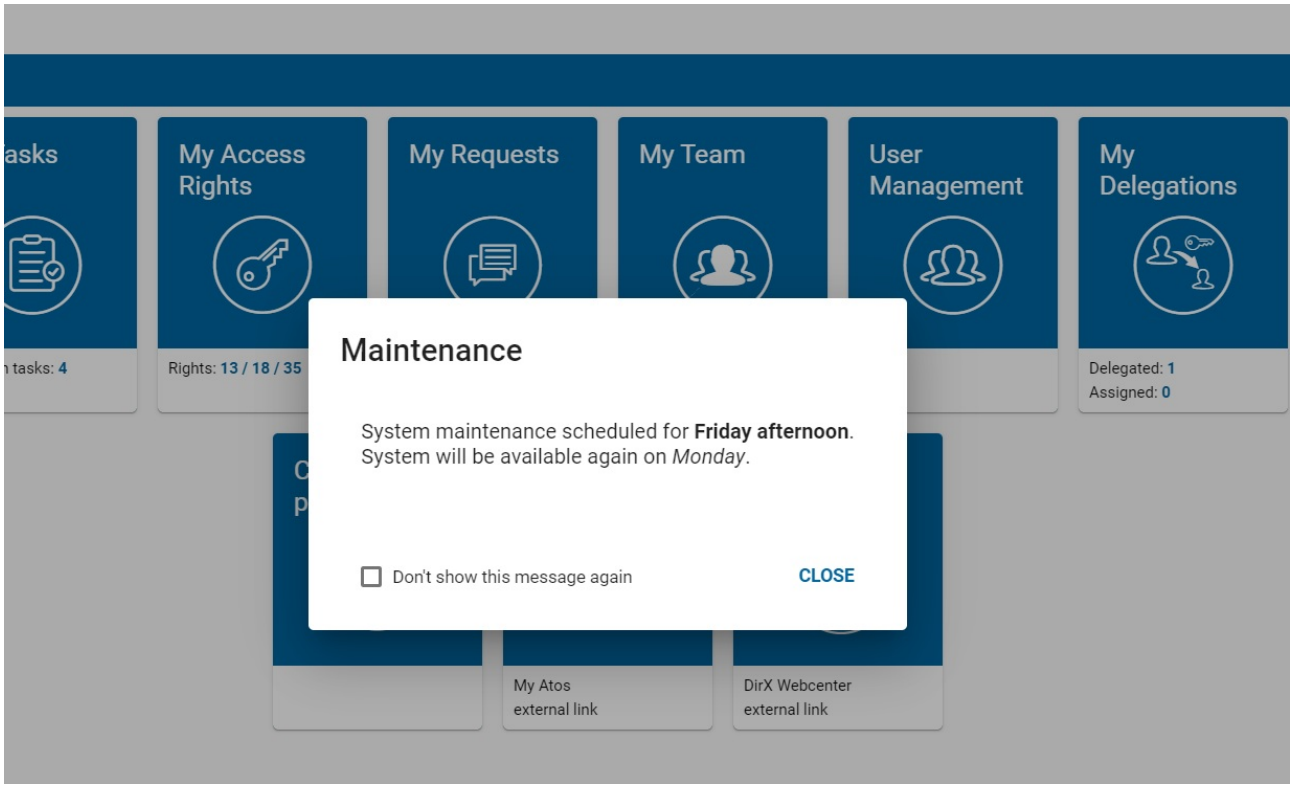

Figure 11. BUI Login Information Dialog

The message can be entered in the **login-message.json** file, which can be found in the config folder with the **config.json** file.

Here is an example of **login-message.json**:

```
{
    "title": "Maintenance",
    "title_DE": " Wartungsarbeiten",
    "message": "System maintenance scheduled for the end of the
week.”,
    "message_DE": Am Freitagnachmittag finden Wartungsarbeiten
statt.",
    "valid-from": "2020-05-01",
    "valid-until": "2020-07-30"
}
```

It is possible to enter different language translations by appending the corresponding country code next to the title and message fields. Users will be shown the message in the preferred language selected in the application.

Messages can have defined periods in which they are active through the **valid-from** and **valid-to** fields. These values are not required; limitations are applied only if they exist.

Users have the option of hiding viewed messages and their choice will be saved in DirX

Identity under the dxrPreferences parameter as shown in the following example:

```
{"BUI":{"1.0":{"preferredThemeFilename":"styles-brown.css"
,"hideLoginMessage":-2084090855}}}
```

The **hideLoginMessage** value stores a hash ID of the title and message. This value is then compared to a computed one from the **login-message.json** file, and if it has changed, the users will be notified again.

Appendix A: Configure the DirX Identity Business User Interface with DirX Access Policy Enforcement (PEP)

This chapter describes all configuration steps required to configure BUI to work with DirX Access Policy Enforcement (DXA PEP).

The DirX Access PEP enable an authentication and authorization layer above any web application. For this scenario, DXA PEP acts as a security layer for DirX Identity REST services application, and the BUI needs to provide a security token (cookie) to access these services.

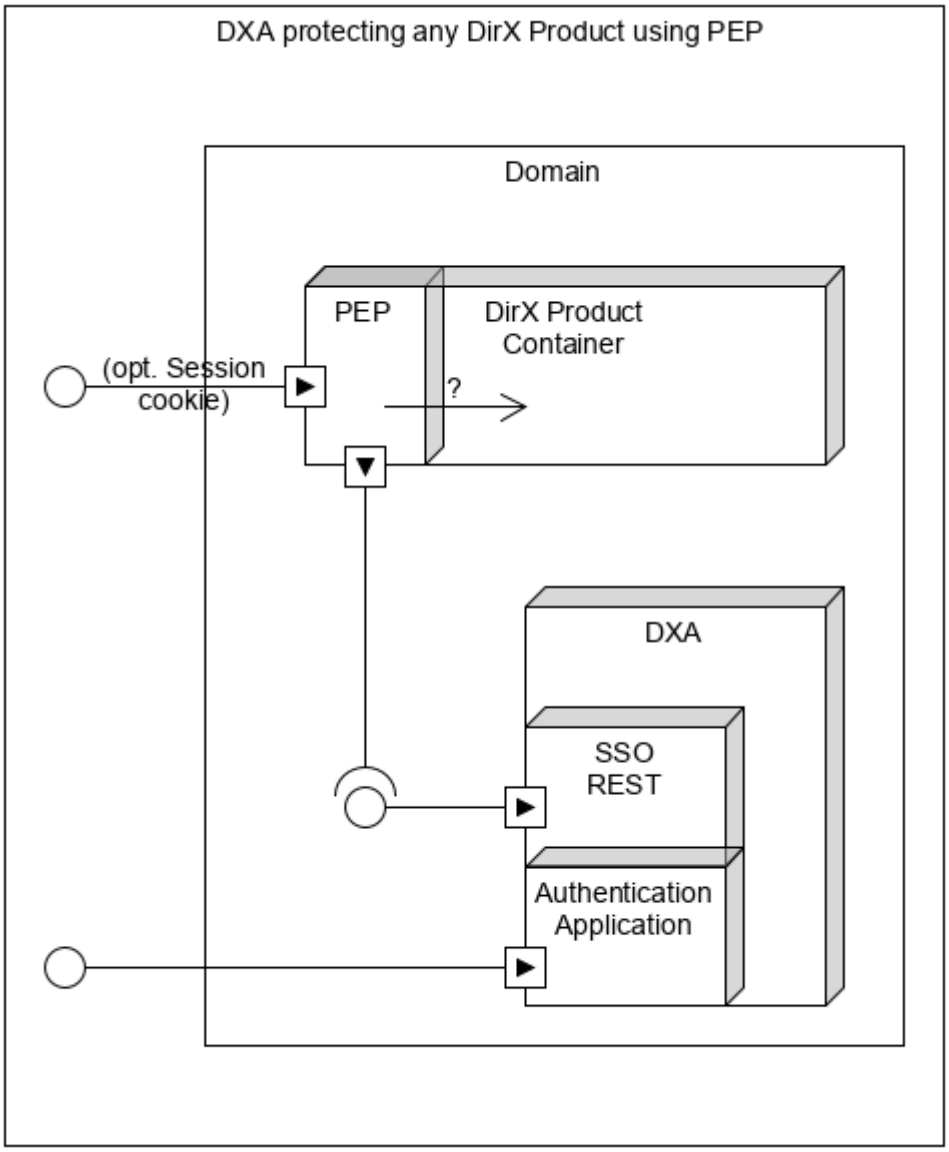

Figure 12. DirX Access PEP Environment

A.1. Configure DirX Access with Policy Enforcement Points (PEP)

This guide is for DirX Access version 9.x and for this use case, DXA PEP is configured only for authentication purposes, because DXI REST services implements its own the authorization layer and authorization features from DXA are not used.

A.1.1. Starting DirX Access Services

Following Windows/Linux services are required for this scenario:

- Start DirX Directory

The DirX Directory must contain DirX Access configuration and a repository with DirX Identity Business User Interface users.

- Start "DirX Access Services *domain*" service (e.g., DirX Access Services My-Company).
- Start "DirX Access *domain* LoadBalancer" service (e.g., DirX Access My-Company LoadBalancer). This service needs to be start if DirX Access is configured with load balancer).
- Start "DirX Access Extensions Apps" - optional - This step is optional and starts additional DXA application that can be used to check and test different DXA features.

A.1.2. Configure DirX Access for PEP

The DXA configuration is available through a web interface, DXA console.

- Open in the web browser (recommended: Google Chrome) the DXA console. The access URL should be similar with:

http(s)://*dxa_server.dxa_domain:dxa_port*/console/ManagementFrameSet

For DXA Demo server this URL is:
http://my-server.my-company.example:10114/console/ManagementFrameSet

- Authenticate to DXA console (default DXA console credentials are: **dirxaccesadmin /dirxaccess**).
- Create and configure port assignments
  - Navigate to **Servers→Port Assignments**
  - Create or duplicate new port assignments
  - Save changes.
- Create a new DXA server or duplicate an existing one
  - Navigate to **Servers→Server list**
  - Create a new DXA server (top-right button, above the server list table)
  - Configure DXA server:

        - **“Identifier** “- mandatory - DXA server identifier
        - “**Description** “- optional - DXA server description text
        - **“Cache IP address (local)”** - mandatory - DXA IP address
        - “**Server hostname”** - mandatory - DXA server hostname
        - **“Port assignment identifiers”** - mandatory - Set port assignment identifiers (from **Port Assignments** list available top-right in **Selectable objects** tab)
- Check and validate users list
    - Navigate to **Users→List**
    - Check if users are available, connection to DirX Directory is working.
    - If the list is not available, check settings in **Configuration→User repository**
- Configure Request injection
    - Navigate to **Subjects→Request injection**
    - Create a new **Request injection** item (top-right button)
    - Set following attributes
        - **“Identifier”** - mandatory - Set to *X_FORWARDED_USER for DXI REST service* value.
        - **“Description”** - optional - Set to “Required field by DXI REST service for authentication”
        - **“Type”** - mandatory - Set to *HttpRequestHeader*
        - **“Keyword”** - mandatory - Set to *X_FORWARDED_USER* - this field will be available to DXI REST service after a successful authentication
    - **“Keyword value separator”** - mandatory - Set to *“=”*
        - **“Request injection value template identifier”** - mandatory - Set to *LoginName*
    - Save or update changes
- Create DirX Access PEP
    - Navigate to **Configuration→Policy Enforcement Pointes→Deployable Web**
    - Create a new PEP or duplicate and existing configuration (for Tomcat 9),
    - Set following values:
        - **“PEP Type”** - mandatory - Set to *Tomcat 9.0*
        - **“Identifier”** - mandatory - Set to *Tomcat 9.0 DXI REST services*. This value will be used as a handler in Tomcat server configuration for connector and valve setup.
        - **“Description”** - optional - Set to *Tomcat 9.0 PEP configuration for DXI REST services 8.10*
    - Open Client settings
        - **Use SSL/TLS** - optional - Set to true to use SSL/TLS encrypted communication between DXA PEP and DXA server.
    - Open **PEP settings**

- **“Authority”** - mandatory - This is a handler for current PEP and server configuration, and format is a string: *domain:port*, (e.g. my-company:8080).
- **“Common extensions to exclude”** - optional - This is a list of web resources excluded from authentication process (e.g. gif, jpg, png, html, etc.).
- **“Other extensions to exclude”** - optional - Extension of the above list. Add *js* and *json* values to this list.
- **“Authorization failed URL”** - mandatory - This URL is used to redirect to a failed authentication. This page must exist to Tomcat server and must display the failed authentication message due to insufficient rights to use the DXA service (this is not and invalid or incorrect username or password message). E.g. */my-company/not_authorized_error.html*.
- **“Request injection templated identifier”** - mandatory - Add *X_FORWARDED_USER for DXI REST services* from the list (select from right tree).

If the identifier is not available in the list, then restart DXA services.

The tree may not work correctly with some web browsers (e.g. Mozilla Firefox).

- Open **Web PEP settings**
- Set following values:
  - **“Enable HTTP cookie”** - mandatory - Set to *true*.
  - **“Cookie name”** - mandatory - Set the cookie name, e.g. *DXAIDMYCOMPANY*
  - **“Cookie domain”** - mandatory - Set the cookie domain, e.g. *.my-company.example*.
  - **“Enable URL rewriting”** - mandatory - Set to *true*.
  - **“Form authentication enabled”** - optional - Set to *true*.
  - **“Do process HTTP request parameters to extract custom data”** - optional - Set to *true*.
  - **“Set no-cache header”** - optional - Set to *true*.
- Open Web PEP deployments settings
- Set following values:
  - **“File system folder for PEP deployment”** - mandatory - This is the path where DXA will deploy the PEP configuration and PEP implementation. These files must be copied (recommended approach) to the Tomcat with DXI REST services configuration: *DXA_PEP_DEPLOYMENT_FOLDER* (e.g. *C:\Program Files\Tomcat90\My-Company BUI*)
  - **“File system folder for client keystore deployment”** - optional/mandatory for SSL - This is the path where DXA will deploy the keystore for SSL communication between PEP and DXA server: *DXA_PEP_DEPLOYMENT_FOLDER/conf* (e.g. *C:\Program Files\Tomcat90\My-Company BUI/conf*).
  - **“Update Apache Tomcat Server configuration”** - mandatory - Set to *true*.

- **“Apache Tomcat version”** - mandatory - Set to *Tomcat90*.

Is preferrable to manually copy the PEP configuration and PEP jar files to the DXI REST services deployment server.

- Save or update changes
- **IMPORTANT**: Deploy (manually copy) the PEP configuration.
- **RECOMMENDED**: Restart DXA services.

A.2. Deploy DXA PEP to Apache Tomcat 9.0

These changes are for DXI REST services application server

- Stop Apache Tomcat (service or application)
- Backup Apache Tomcat *conf* folder
- Backup Apache Tomcat *lib* folder
- Copy to Apache Tomcat *lib* folder, the PEP jar files from location where DXA deployed the PEP implementation: *DXA_PEP_DEPLOYMENT_FOLDER>\lib* (e.g. *c:\Program Files\Tomcat90\My-Company BUI\lib*)

Apache Tomcat *lib* folder, not to DXI REST services lib folder.

- Copy to Apache Tomcat *conf* folder the PEP configuration files from location where DXA deployed the PEP configuration: *DXA_PEP_DEPLOYMENT_FOLDER>\conf* (e.g. *c:\Program Files\Tomcat90\My-Company BUI\conf*)
- Navigate to Apache Tomcat *conf* folder and open **server.xml** file.
- Search for *ClientId* keyword and check and/or change values:
  - **“ClientID”** - This id must be the same with DXA PEP configuration.
  - **“ClientKeystoreFilename”** - Check if path to keystore is correct.
  - **“ServerPrimaryHost”** - DXA server host name, check if is correct.
  - **“ServerPrimaryPort** “- DXA server port, check if is correct.
  - **“ServerPrimaryPortSSL”** - DXA SSL server port, check if is correct.
- Search for *Connector* keyword and check and/or change values:
- **“keystoreFile”** - Check if path to keystore is correct.
- Save changes to **server.xml**.
- Don’t Apache Tomcat yet.

A.3. Configure DirX Identity REST Services with DXA PEP

The DXI REST services requires to enable security configuration for DXA PEP. This file is available in DXI REST services deployment folder:

*dxi_install_path***restServices\DirXIdentityRestServices\DirXIdentityRestService-*_domainy_*\WEB-INF\authenticationSamples**

- Stop Apache Tomcat (service or application)
- Navigate to DXI REST services deployment folder (e.g. *dxi_install_path*\restServices\DirXIdentityRestServices\DirXIdentityRestService-*domainy*\WEB-INF).
- Backup **security.xml**.
- Copy *authenticationSamples***security-dirxaccess.xml** file to *WEB-INF* folder and rename it to **security.xml**.
- Open **security.xml** and check *principalRequestHeader* value, must be *X_FORWADED_USER*.

This value is set in DXA PEP configuration, to **Request injection**, the name from DXA PEP and from **security.xml** must be same.

- Save **security.xml** changes
- Backup **security.properties**.
- Open *security.properties* and comment line:
- *auth.userDetails.valuePattern = ^CN=(.*?),".*
- Save *security.properties* changes.
- Start Apache Tomcat (service or application).

A.4. Configure DXI Identity Business User Interface with DXA PEP

These settings are for communication between BUI and DXI REST services, DXA PEP configuration is available in server.xml in Apache Tomcat.

- Open Business User Interface configuration file: *config.json*.
- Search for *dxaPepAuthServer* section.
- Set communication protocol between DXI REST services and BUI: *http* or *https*.
- Set DXI REST services host name (e.g. *dxi-server.my-company.example*)
- Set DXI REST services port (e.g. *8080*)
- Set DXI REST services domain (e.g. */DirXIdentityRestService*)

- Set DXI Logout path (e.g. *http://dxi-server.my-company.example:8080/my-company/logoutAction*)
- Enable DXA PEP as authentication option
- In **Security→Authentication** and *DXA_PEP* to available list.

Position in authentication methods list is considered. If *DXA*_PEP is the last option, DXA PEP authentication method will be listed at the end of the available authentication options in the Business User Interface login page.

- Reload Business User Interface application (refresh the page).

Appendix B: Configure the DirX Identity Business User Interface with OAuth2 (OIDC/PKCE)

B.1. Start DirX Access Services

- Start DirX Directory service (if is necessary).
- Start DirX Access Service *domain* service (e.g. DirX Access Services My-Company).
- Start DirX Access *domain* LoadBalancer service (only if DirX Access is configured with Load Balancer feature enabled).
- Start DirX Access Extensions Apps (optional).

B.2. Configure DirX Access for OAuth2 Protocol

This chapter describes how to configure DirX Access 9.x product (with SP1) for OAuth2 protocol. From OAuth2 protocol will use Open ID Connector with PKCE protocol extension.

Follow these steps:

- Open in a web browser the DXA console interface (e.g., http://my-server.my-company.example:10114/console/ManagementFrameSet).
- Authenticate to DXA console (default credentials are: dirxaccessadmin/dirxaccess)
- Create and configure port assignments - this step will create port used by DirX Access to provide services (authentication and authorization) to other applications.
  - Navigate to **Servers→Port Assignments**.
  - Create or duplicate a new **Port assignment** (top-right button).
  - Save changes
- Create a new DirX Access server or use an existing one
  - Navigate to **Servers→Server list**
  - Create a new **server** (top-right button)
  - Configure DXA server:
    - **“Identifier”** - this is the DXA server identifier and is used to identify this server through the DXA settings.
    - **“IP address”** - this is the DXA server IP address
    - **“Server hostname”** - this is the DXA hostname server. This value should be used in any connection settings and security settings (e.g., server certificates).
    - **“Port assignment” -** this port assignments are created and assigned to current server, from **Port Assignments** settings.
  - Save changes

- Check Users list from a connection to an LDAP server.
  - Navigate to **Users→List**
  - heck if users are available, and connection to DirX Directory works.
  - If users are not available check settings in **Configuration→User repository**
- Configure Client metadata
  - Navigate to **Federation→OAuth→Client metadata**.
  - Create new Client metadata (top-right button)
  - Configure Client metadata:
    - **“client_id”** - client_id value will be used by the OAuth2 client application, e.g., for BUI can be **dxiBUIApplication.**
    - **“client_type”** - this defines where the OAuth2 client application is running. For almost all SPA web applications, this value must be set to **public**, because the web browser is not considered a trusted environment to run an application.
    - **“token_endpoint_auth_method”** - set this value to **none**. For other options, consult DXA official documentation.
    - **“redirect_url”** - this URL is used by the OAuth2 server to redirect back to the client after the authentication process is completed (with success or not). The client application must be able to handle this redirect and detect is authentication was successful (OpenID token is available and valid).
    - **“grant_types”** - this are grant type required by OAuth2 OpenID protocol and must be: **authorization_code**, **implicit** and **refresh_token**. For other available options, consult OpenID protocol specifications.
    - **“response_type” -** this are responses required by OAuth2 OpenID protocol and must be: **code**, **token**, **id_token**, **id_token_token**. For other available options, consult OpenID protocol specifications.
    - **“scope”** - this are scopes requested by the client. For this scenario only **openid** and **offline_access** are required, but if is necessary this **scope** can be extended with other values (profile, email, etc.).
  - Save changes

Example for a Client metadata configuration for DirX Identity Business User Interface:

```
{
   "client_id": "dxiBUIApplication",
   "client_type": "public",
   "token_endpoint_auth_method": "none",
   "redirect_uris": [
      "https://bui.my-company.example:4123/login"
   ],
   "grant_types": [
      "authorization_code",
```

```
      "implicit",
      "refresh_token"
   ],
   "response_types": [
      "code",
      "token",
      "id_token",
      "id_token_token"
   ],
   "scope": "openid offline_access"
}
```

- Configure **Client endpoints** (optional)
  - Navigate to **Federation → OAuth → Client endpoints**
  - Create a new or modify current Client endpoint (top-right button).
  - Go to **OAuth client endpoint** settings.
  - Change "**OAuth authentication method**" to **OAuth Authentication**.
  - Change "**Default partnership (client) metadata**" to Client metadata configuration created above (e.g., **dxiBUIApplication**)
  - Save changes.
- Configure **Keystore**
  - Navigate to **Key management → Features → Generate Sample**.
  - Set "**Identifier**" (e.g., **dxiBUIOAuthId**).
  - Set "**Password**" (e.g., **dirx**).
  - "**Client Verification**" - set **Sample type** to **Client Verification**.
  - Generate keystore.
- Configure **Subjects**
  - Navigate to **Subjects → OAuth token → OAuth Attribute template Value Construction**.
  - Create a new template (top-right button)
  - Set "**Identifier**" (e.g., **buiLoginNameValue**).
  - Set "**Return type**" to **STRING**.
  - Set "**Container**" to **SSO_SERVICE**.
  - Set "**Abstraction**" to **AUTHENTICATION_INFO**.
  - Set "**Detail**" to **AUTHENTICATION_CN**.

This field defines which information will be provided to REST service inside the authentication token. In this example is **CN**.

- Save changes.
- Navigate to **Subjects → OAuth token → OAuth Attribute template Construction.**
- Create new template (top-right button).
- Set "**Identifier"** (e.g., **dxiBUIOAuthId**).
- Set Name (e.g., l**oginName**).

This field will be available in access token with **CN** of the authenticated user). This field must be processed by DXI REST service as authentication principal.

- Set "**Mandatory"** to **true**.
- Set **"OAuth attribute value template" -** this template was created in the previous step. (e.g., **buiLoginNameValue**).
- Save changes.

- Configure OAuth2 server
  - Navigate to **Federation → OAuth → Servers**.
  - Create new **OAuth2 server**. (top-right button)
  - Configure OAuth server:
  - Basic settings:
    - **"identifier" -** set configuration identifier (e.g., **dxiOAuthServer**).
    - **"description"** - optional - set configuration description set description
    - **"OAuth server endpoint"** settings:
      - check "**issuer"** (e.g., https://my-server.my-company.example:11115/auth)
      - select "**client metadata"** (from right tree), and add newly created client metadata (e.g., **dxiBUIApplication**).
      - set "**HTTP redirect URI support**" to **true**.
      - set" **Invalid client metadata support**" to **true** for debug.
      - set "**Keystore identifier**" (e.g., **dxiBUIOAuthId**).
      - set "**Keystore password**" (e.g., **dirx**).
      - set "**Signing key alias**" to **federation-sign.dirxaccess**
      - set "**Signing key password**" (e.g., **dirx** or **dirxaccess**).
  - Basic configuration
    - set "**OAuth support**" to **true**.
    - set "**OpenID Connect support**" to **true**.
  - Authorization and Token endpoint configuration
    - set "**Grant types**" to: **authorization_code**, **implicit** and **refresh_token**.

These values must match and be used the OAuth2 OpenID client

application.

- set “Scopes” to: offline_access and openid.

These values must match and be used the OAuth2 OpenID client application.

- set values:
  - (optional) adjust “**Authorization code validity”** in seconds.
  - (optional) adjust “**Access token validity”** in seconds.
  - **IMPORTANT** set “**Access token representation”** to **JWT.**
  - set “**Refresh token”**.
  - set “**Attribute templates”** to newly created configuration template (e.g., **dxiBuiOAuth**).
  - set “**JWE signing key representation”** to **None**.
  - **IMPORTANT** set “**PKCE for OAuth 2.0 support”** to **true**.

Make sure you installed the latest Service Pack for DirX Access 9.x (default version 9.0 does not work correctly with PKCE extension).

- Save changes.
- (optional) Check server metadata.

Example for a Server metadata configuration from:

```
https://my-server.my-company.example:11115/auth/.well-known/oauth-
authorization-server

{
  "issuer": "https://my-server.my-company.example:11115/auth",
  "authorization_endpoint": "https://my-server.my-
company.example:11115/auth/authz",
  "token_endpoint": "https://my-server.my-
company.example:11115/auth/token",
  "jwks_uri": "https://my-server.my-company.example:11115/auth/.well-
known/jwks.json",
  "scopes_supported": [
  "openid",
  "offline_access"
  ],
  "response_types_supported": [
```

```
  "id_token token",
  "code",
  "id_token",
  "token"
  ],
  "response_modes_supported": [
  "fragment",
  "query"
  ],
  "grant_types_supported": [
  "refresh_token",
  "implicit",
  "authorization_code"
  ],
  "token_endpoint_auth_methods_supported": [
  "client_secret_post",
  "none",
  "client_secret_basic"
  ],
  "subject_types_supported": [
  "public"
  ],
  "id_token_signing_alg_values_supported": [
  "HS256",
  "HS512",
  "RS256",
  "HS384"
  ],
  "display_values_supported": [
  "page"
  ],
  "claim_types_supported": [
  "normal"
  ],
  "claims_parameter_supported": false,
  "request_parameter_supported": false,
  "request_uri_parameter_supported": false,
  "require_request_uri_registration": false
}
```


- Configure Provider endpoints

- Navigate to **Federation → OAuth → Provider endpoint**.
  - Create a new Provider endpoint.
- Set identifier for **provider**.
  - Set **description** (optional) for provider
  - Set **OAuth server** (e.g., **dxiOAuthServer**).
  - Set/check **context path** (e.g., **/auth**).
  - Set/check **Primary port assignment identifier** (e.g., **11115**)
  - Check **Token endpoint**.
  - Save changes.

- Create a new **Policy rule** - IMPORTANT
  - Navigate to **Policies → RBAC Authentication → Rules**.
    - Create a new **rule**. (top-right button)
    - Set **Identifier** (e.g., **dxiBUIRules**).
    - Set **Description** (optional).
    - Set Actions (e.g., **Any Action**).

      

      Selected action: "Any Action" is to general, and should be changed to a more restricted value.

    - Save changes.
    - An authenticated user should be allowed to access DXA resources
    - Navigate to **Policies → RBAC Authorization → Policies**
    - Set **Identifier** to **User.**
    - Add **dxiBUIRules** as active rule.
    - Save changes.

B.3. Configure DXI REST Service

B.4. Configure BUI Application

Configuration settings are available in the **config.json** file, located in the Business User Interface installation folder *install_path*/**web/BusinessUserInterface-***domain*/**assets/config/**. The content of this file is case sensitive.

Set following OAuth2 parameters in "**oauth2AuthServer**" entry:

- "**clientId**" - the application ID used in OAuth2 Identity Provider configuration. The Business User Interface application must be registered to Identity Provider with this id.
- "**issuer**" - the URL to OAuth2 provider.

Issuer examples for different OAuth2 providers:

- for DirX Access:
  `https://my-server.my-company.example:11115/auth`
- for RedHat Keycloak:
  `http://localhost:8180/auth/realms/my-company`

Appendix C: Configure OAuth2 for DirX Identity Business User Interface and REST Service with Red Hat Keycloak IdP

C.1. Configure Red Hat Keycloak

All the settings described below are against RedHat Keycloak version 21.1 and are recommended settings for development and testing and should not be used in any production environment. All settings are suggested and for production environment consult official documentation available on product website.

C.1.1. Create a Keycloak realm

This is like a domain in DirX Identity and defines boundaries and settings for a IdP domain.

Authenticate to Administrative Console with an administrative account and create a new realm. Suggested realm name is *my-company*, enable this realm and click Create button.

C.1.2. Setup User federation

Switch to newly created realm.

Select **User federation** and click “Add new provider”, select “LDAP” as provider.

Provide a name (e.g DirX Directory), and at “Vendor” select “Other” from list.

- Set “Connection URL” to DirX Directory server (e.g., ldap://DXIPTEST01-VM:389)

If highly recommended to evaluate all other options (Enable StartTLS, Use Trusted SPI, etc).

Click “Test connection” button to test connection to LDAP server.

- **Bind type** - set as “simple”.
- **Bind DN** - set to a proper DN for a admin user (e.g. cn=DomainAdmin,cn=My-Company).
- **Bind credentials** – set authentication password for access to DirX Directory.

Click “Test authentication” button to test authentication against DirX Directory.

LDAP searching and updating

- **Edit mode** - set as “READ_ONLY”.
- **User DN** – set DN to Identity Users container (e.g. cn=Users,cn=My-Company)
- **Username LDAP attribute** - set to “cn”.
- **RDN LDAP attribute** – set to “dn”.

- **UUID LDAP attribute** – set to “dirxEntryUUID”.
- **User object classes** – leave unchanged. Default value is “inetOrgPerson, organizationalPerson”.
- **User LDAP filter** – leave it empty or adjust according to requirements.
- **Search scope** – select “Subtree” value.
- **Read timeout** – leave it empty or adjust according to requirements.
- **Pagination** – set to “Off” or adjust according to requirements.

Synchronization settings

- **Import users** – set to “Off”
- **Sync Registration** – set to “Off”
- **Batch size** – leave empty
- **Periodic full sync** – set to “Off”
- **Periodic changed users sync** – set to “Off”

Kerberos integration

- Everything set to “Off”

Cache settings

- **Cache policy** – set to “DEFAULT”

Advanced settings

- Everything set to “Off”.

Click “Save” button to save changes.

C.1.3. Setup Realm settings

- Select **General** tab.
  - **Realm ID** – should be already set to a value provided on create realm action (e.g., my-company).
  - **Display name** – optional – plain text for realm name.
  - **HTML Display name** – optional – HTML text for realm name.
  - **Frontend URL** – optional – leave empty.
  - **Required SSL** – “External requests”,
  - **ACT to LoA Mapping** – optional – no entries.
  - **User-managed access** – set to “Off”.

Highly recommended to follow the “OpenID Endpoint Configuration” link from **Endpoints** and evaluate the content from this link. This content is used by OAuth2 client during the

discovery process.

Click “Save” button to save changes.

- Select **Login** tab.
  All values should be set to “Off”
- Select Email tab.
  Adjust values according to requirements.
- Select **Themes** tab.
  Adjust values according to requirements.
- Select **Keys** tab.
  Use default values or adjust values according to requirements.
- Select **Events** tab.
  Use default values or adjust values according to requirements.
- Select **Localization** tab.
  Use default values or adjust values according to requirements.
- Select **Security defences** tab.
  Use default values or adjust values according to requirements.
- Select **Sessions** tab.
  Use default values or adjust values according to requirements.

Here can be change the Offline Session tokens lifespan.

- Select **Tokens** tab.
  Use default values or adjust values according to requirements.

Here can be change the Session tokens lifespan and default signature algorithm, should be “RS256”.

- Select **Client policies** tab.
  Use default values or adjust values according to requirements.
- Select **User registration** tab.
  Use default values or adjust values according to requirements.

C.1.4. Setup a client

Click “Create client” button

- **Client type** - select “OpenID Connect”.
- **Client ID** – set client name (e.g., dxibui).
- **Name** – display name for client (e.g., DirX Identity Business User Interface).
- **Description** – description of the client.
- **Always display in UI** – set to “On”

Click “Next” button to continue.

- **Client authentication** – IMPORTANT – set to “Off”.
- **Authorization** – set to “Off”.
- **Authentication flow** – set “Standard flow” and “Direct access grants” to “On”, everything else set to “Off”

Click “Next” button to continue.

- **Root URL** – optional – set this value according to requirements.
- **Home URL** – optional – set this value according to requirements.
- **Valid redirect URL** – set this value according to requirements (e.g., http://localhost:8080/BusinessUserInterface-My-Company/*.
- **Valid post logout redirect URIs** – set this value according to requirements (e.g., for tests put + character).
- **Web origins** – set this value according to requirements (e.g., for tests put + character).

Click “Save” button to save changes.

Select client for the list and review the settings in **Settings** tab.

- Select **Roles** tab.
  - Leave empty.
- Select **Client scopes** tab.
  - **acr** - set to “Default”
  - **offline_access -** set to “Default”
  - **profile** – set to “Default”

  Everything else set to “Optional”.
- Select **Sessions** tab.
  - Should be empty
- Select **Advanced** tab.
  - Change values according to requirements.
  - **Proof Key for Code Exchange Code Challenge Method** – IMPORTANT – set to “S256”

Click “Save” button to save changes.

C.1.5. Select a Client scope (Audience)

Select **Client scopes** menu entry.

Click “Create client scope” button.

- **Name** – set a name for client scope (e.g., DXI_Audience)
- **Description** – provide a description for client scope.

- **Type** – set to “Default”.
- **Protocol** – set to “OpenID Connect”.
- **Display on content screen** – set to “On” or “Off”
- **Consent screen text** – optional – provide a text for consent screen.
- **Include in token scope** – IMPORTANT - set to “On”
- **Display Order** – leave empty.

Click “Save” button to save changes.

Select newly created client scope from list.

- Select **Mappers** tab.
  - Click “Add mapper” button and select “By Configuration” option.
  - Select “Audience” from list.
  - **Mapper type** – set to “Audience”.
  - **Name** – name of mapper (e.g., DXI_audience).
  - **Included Client Audience** – IMPORTANT – select from list the client (e.g., dxibui).
  - **Included Custom Audience** – leave empty.
  - **Add to ID token** – set to “Off”.
  - **Add to access token** – IMPORTANT – set of “On”

Click “Save” button to save changes.

Select **Clients** menu entry.

Click to the client (e.g., dxibui) from list.

Select tab **Client scopes.**

Click **Add client scope**, select client scope from the list, select **Default** and confirm changes.

Click “Save” button to save changes.

Now Keycloak should be ready for use for OAuth2-OIDC-PKCE setup.

C.2. Configure DirX Identity REST Services

- Stop Apache Tomcat server
- Locate **security.properties** file inside DirX Identity restServices installation folder.
  - Go to **OAUTH2 / JWT authentication** section.
  - Set **auth.jwt.authenticationClaimAudiences** to values from audicences map for client scope (e.g., dxibui)

- Check value for **auth.jwt.authenticationClaimName**, default value is set to preferred_username.
- Check value for all other entries. Default values are set to search for enabled users in DXI user container and use the CN as value select, search users by cn.
- Save the file.

- Locate **security-oauth2.xml** file inside **authenticationSamples** folder.
  - Inside file, find **jwk-set-uri** entry, and replace "place here jwk-set-uri url" text with the URL to Keycloak server. This value can be obtained by going to <realm-name> → Realm Settings menu → Endpoints → OpenID Endpoint Configuration (e.g., jwt-set-uri= "http://dxiptest01-vm:9090/realms/my-company/protocol/openid-connect/certs")
  - Save the file.

There are two variants:

Variant 1) Copy **security-oauth2.xml** to the upper folder with a new name **security.xml** and replace existing **security.xml** file.

Variant 2) Merge content of **security-oauth2.xml** with content of the **security.xml** file for above folder.

- Start Apache Tomcat server

C.3. Configure DirX Identity Business User Interface

all settings are case-sensitive.

- Locate **config.json** file inside DirX Identity BUI installation folder.
- Go to **oauth2AuthServer** section
  - **protocol –** set to “http” or “https”
  - **host** – set host name of the rest services (e.g., “DXIPTEST01-VM”)
  - **port** – set port for the rest services (e.g., “8080”)
  - **domain** – set the rest services DXI domain (e.g., “/DirXIdentityRestService-My-Company”
  - **clearHashAfterLogin** – leave to true
  - **clientId** – IMPORTANT **–** set the client id configured in Keycloak (e.g., “dxibui”)
  - **issuer** – IMPORTANT – set the issuer URL. This value can be obtained by going to <realm-name> → Realm Settings menu → Endpoints → OpenID Endpoint Configuration (e.g., http://dxiptest01-vm:9090/realms/my-company).
  - **redirectUri** – IMPORTANT – “/main”. This value must match with settings from Keycloak
  - **requireHttps** – IMPORTANT – For tests this value can be set to false, for production

this must be always true.

- **scope** – “openid offline_access” – “offline_access is required for token refresh action
- **sessionChecksEnbled** – set to false,
- **showDebugInformation** – set to true,
- **silentRefreshUri** – not used.
- **timeoutFactor** – default value in 0.75. This value represents a percent and is used to refresh an access token. As example, for default value, when ¾ of access token validity time is passed and a new access token is requested from Keycloak.
- **useHashLocationStrategy** – for now only option is true
- **useSilentRefresh** – set to false

- Go to **security** section
  - **authentication -** add “OAUTH2” to the list. Order is important.
- Save the file and reload Business User Interface in browser.

DirX Product Suite

The DirX product suite provides the basis for fully integrated identity and access management; it includes the following products, which can be ordered separately.

DirX Identity

DirX Identity provides a comprehensive, process-driven, customizable, cloud-enabled, scalable, and highly available identity management solution for businesses and organizations. It provides overarching, risk-based identity and access governance functionality seamlessly integrated with automated provisioning. Functionality includes lifecycle management for users and roles, cross-platform and rule-based real-time provisioning, web-based self-service functions for users, delegated administration, request workflows, access certification, password management, metadirectory as well as auditing and reporting functionality.

DirX Directory

DirX Directory provides a standards-compliant, high-performance, highly available, highly reliable, highly scalable, and secure LDAP and X.500 Directory Server and LDAP Proxy with very high linear scalability. DirX Directory can serve as an identity store for employees, customers, partners, subscribers, and other IoT entities. It can also serve as a provisioning, access management and metadirectory repository, to provide a single point of access to the information within disparate and heterogeneous directories available in an enterprise network or cloud environment for user management and provisioning.

DirX Access

DirX Access is a comprehensive, cloud-ready, scalable, and highly available access management solution providing policy- and risk-based authentication, authorization based on XACML and federation for Web applications and services. DirX Access delivers single sign-on, versatile authentication including FIDO, identity federation based on SAML, OAuth and OpenID Connect, just-in-time provisioning, entitlement management and policy enforcement for applications and services in the cloud or on-premises.

DirX Audit

DirX Audit provides auditors, security compliance officers and audit administrators with analytical insight and transparency for identity and access. Based on historical identity data and recorded events from the identity and access management processes, DirX Audit allows answering the “what, when, where, who and why” questions of user access and entitlements. DirX Audit features historical views and reports on identity data, a graphical dashboard with drill-down into individual events, an analysis view for filtering, evaluating, correlating, and reviewing of identity-related events and job management for report generation.

For more information: support.dirx.solutions/about

Legal remarks

On the account of certain regional limitations of sales rights and service availability, we cannot guarantee that all products included in this document are available through the Eviden sales organization worldwide. Availability and packaging may vary by country and is subject to change without prior notice. Some/All of the features and products described herein may not be available locally. The information in this document contains general technical descriptions of specifications and options as well as standard and optional features which do not always have to be present in individual cases. Eviden reserves the right to modify the design, packaging, specifications and options described herein without prior notice. Please contact your local Eviden sales representative for the most current information. Note: Any technical data contained in this document may vary within defined tolerances. Original images always lose a certain amount of detail when reproduced.