

Inštalácia a konfigurácia pre staršie verzie D2000



Táto kapitola obsahuje postup inštalácie SmartWeb aplikácie relevantný pre D2000 do verzie 12.0.61 vrátane, kedy sa pre Smart Web nasadzoval s aplikaným serverom [Wildfly AS](#).

- [Inštalácia na Windows OS](#)
 - [Inštalácia JRE 1.8](#)
 - [Inštalácia Wildfly AS a SmartWeb aplikácie](#)
- [Inštalácia na Linux OS \(Ubuntu\)](#)
 - [Inštalácia JRE 1.8](#)
 - [Inštalácia Wildfly AS a SmartWeb aplikácie](#)
 - [Použitie portov 80/443 pre Wildfly server na Linuxe](#)
- [Upgrade SmartWeb aplikácie](#)
- [Konfigurácia Wildfly AS pre SmartWeb](#)
 - [Základná konfigurácia standalone.xml](#)
 - [Odporúčaná konfigurácia pre optimálny beh a zabezpečenie servera](#)
 - [Voliteľná konfigurácia silného šifrovania pre HTTPS](#)
 - [Voliteľná konfigurácia automatického presmerovania HTTP na HTTPS](#)

Inštalácia na Windows OS

Inštalácia JRE 1.8

SmartWeb platforma je implementovaná v Java EE a preto potrebuje na svoj beh aj nainštalované a nakonfigurované Java Runtime Environment verzie 8.

Postup inštalácie je nasledovný:

1. Stiahnu a nainštalova najnovšie JRE 1.8 z oracle.com.
2. Nastavi systémový environment premennú JAVA_HOME na root adresár kde bola JRE nainštalovaná.
3. Pre použitie silných šifier v HTTPS protokole je potrebné stiahnu [Java Cryptography Extension \(JCE\)](#) súbory a prekopírova ich do lib/security adresára JRE. Ak nie je potrebné konfigurova HTTPS protokol tento krok je možné vynecha.

Inštalácia Wildfly AS a SmartWeb aplikácie

Postup pri inštalácii aplikovaného servera Wildfly je je nasledovný:

1. **Rozbali inštalané súbory** v inštalanom adresári D2000 - D2000_EXE/web sa nachádzajú zazipované inštalané súbory aplikovaného servera Wildfly (wildfly.zip) ako aj Java EE aplikácie SmartWeb (smartweb.zip) a tutorial javascriptovej aplikácie (tutorial-application.zip). Všetky tieto archívy je potrebné rozpakovať do toho istého adresára.
2. **Nainštalova aplikovaný server Wildfly** spustením inštalaného skriptu D2000_EXE/web/installWildfly.bat sa automaticky nainštaluje aplikovaný server ako Windows Service s menom: D2000 SmartWeb.
3. **Nainštalova SmartWeb Java EE aplikáciu** spustením inštalaného skriptu D2000_EXE/web/installSmartWeb.bat sa automaticky deployne (nakopíruje) SmartWeb aplikácia aj s preddefinovanou konfiguráciou javascriptovej tutorial aplikácie na aplikovaný server.
4. **Naimportova do D2000** proces SELF.DCS so základnou konfiguráciou D2Connector a konfiguráciu objektov pre javascriptovú tutorial aplikáciu z adresára D2000_EXE/web/install_files/D2000.
5. **Spusti SELF.DCS** (D2Connector) a aplikovaný server Wildfly (cez service.exe aplikáciu)
6. **Otvori tutorial aplikáciu v prehliadači** napr. adrese <http://localhost:8080/smartWeb>.

Inštalácia na Linux OS (Ubuntu)

Inštalácia JRE 1.8

SmartWeb platforma je implementovaná v Java EE a preto potrebuje na svoj beh aj nainštalované a nakonfigurované Java Runtime Environment verzie 8. Na linuxe sa inštaluje príkazmi cez package manager pod root používateľom. Pre Ubuntu distribúciu sú príkazy nasledovné :

```
add-apt-repository ppa:webupd8team/java # nalinkovanie repozitára s java balíkami
apt-get update
apt-get install oracle-java8-set-default
```

Pre použitie silných šifier v HTTPS protokole je potrebné nainštalovať aj [Java Cryptography Extension \(JCE\)](#) rozšírenie nasledovným príkazom:

```
apt install oracle-java8-unlimited-jce-policy
```

Inštalácia Wildfly AS a SmartWeb aplikácie

Postup pri inštalácii aplikovaného servera Wildfly je nasledovný:

1. **Rozbali inštalované súbory** v inštalovanom adresári `/opt/D2000/web` (alebo v inom, závisí od miesta inštalácie D2000, resp. nakopírovania adresára `web`). V tomto adresári sa nachádzajú zazipované inštalované súbory aplikovaného servera Wildfly (`wildfly.zip`) ako aj Java EE aplikácie SmartWeb (`smartweb.zip`) a tutorial javascriptovej aplikácie (`tutorial-application.zip`). Všetky tieto archívy je potrebné rozpakovať do toho istého adresára, napríklad príkazom:

```
cd /opt/D2000/web
unzip *.zip -d .
```

2. **Vytvorí používateľa wildfly.** Kvôli bezpečnosti je vhodné aby Wildfly na Linuxe bežal pod samostatným používateľom ktorého je potrebné najskôr vytvoriť. V nasledujúcom skripte vytvoríme grupu a používateľa `wildfly`, a priradíme ho ako ownera adresára `/opt/D2000/web`.

```
groupadd -r wildfly
useradd -r -g wildfly -d /opt/D2000/web -s /sbin/nologin wildfly.
chown -R wildfly:wildfly /opt/D2000/web
```

3. **Nainštalovať aplikovaný server Wildfly** spustením inštalovaného skriptu `/opt/D2000/web/installWildfly.sh` s parametrami jednoslovný identifikátor služby (napr `DEMO`) a port offset (posunutie portov pre prípad že na serveri beží viacero aplikovaných serverov). Spustenie skriptu je možné vykonať nasledovne:

```
cd /opt/D2000/web
chmod +x *.sh
./installWildfly.sh DEMO 0
```

4. **Nainštalovať SmartWeb Java EE aplikáciu** spustením inštalovaného skriptu `/opt/D2000/web/installSmartWeb.sh` sa automaticky deployne (nakopíruje) SmartWeb aplikácia aj s preddefinovanou konfiguráciou javascriptovej tutorial aplikácie na aplikovaný server.
5. **Naimportovať do D2000 proces SELF.DCS** so základnou konfiguráciou D2Connectora a konfiguráciu objektov pre javascriptovú tutorial aplikáciu z adresára `/opt/D2000/web/install_files/D2000`.
6. **Spusti SELF.DCS (D2Connector) a aplikovaný server Wildfly** cez príkaz:

```
systemctl start wildfly-DEMO
```

7. **Otvorí tutorial aplikáciu v prehliadači** napr. adrese <http://localhost:8080/smartWeb>.

Použitie portov 80/443 pre Wildfly server na Linuxe

Všeobecný rozdiel medzi konfiguráciou Wildfly na Linuxe a Windows je ten že na Linuxe nemôže proces aplikovaného servera obsadiť priamo porty 80/443 (HTTP/HTTPS) kvôli security. Rieši sa to presmerovaním default portov 8080/8443 cez IPTABLES nasledovne:

```
Iptables -t nat -A PREROUTING -i eth0 -p tcp -dport 80 -j REDIRECT --to-port 8080
Iptables -t nat -A PREROUTING -i eth0 -p tcp -dport 443 -j REDIRECT --to-port 8443
```

Upgrade SmartWeb aplikácie

SmartWeb aplikácia sa inštaluje a upgraduje ako štandardná Java EE aplikácia na aplikovaný server Wildfly. Postup je nasledovný:

1. Vypnú aplikovaný server Wildfly AS.
2. Rozbali patchovaný inštalovaný balík `D2000_EXE/web/smartweb.zip`. SmartWeb sa distribuuje v rámci patchov až od verzie D2000 11.2.57.
3. Nakopírovať súbor `install_files/smartWeb.war` do adresára aplikovaného servera Wildfly: `D2000_EXE/web/wildfly13/standalone/deployments`.
4. Premenovať súbor `smartWeb.war` na `<xxx>.war`, kde `<xxx>` môže byť ubovoné meno vystihujúce názov aplikácie. Pod týmto menom bude aplikácia dostupná po spustení aplikovaného servera: na URL ceste <http://<serverDomena.sk>/xxx>.
5. Spusti aplikovaný server Wildfly AS.



Premenovaním archívu aplikácie (`smartWeb.war`) definujeme URL cestu k spustenej aplikácii. Zároveň tým umožníme spustenie viacerých, na sebe úplne nezávislých verzií SmartWeb aplikácií na jednom Wildfly serveri, z ktorých každá bude dostupná na inej URL ceste. Tieto aplikácie sa môžu pripájať na rôzne D2000 inštancie dokonca aj rozdielnych verzií.

Pozor, pre správne fungovanie aplikácie je nevyhnutné názov `.war` súboru aplikácie premenovať iba s použitím alfanumerických znakov a bez diakritiky.

Konfigurácia Wildfly AS pre SmartWeb

Základná konfigurácia standalone.xml

Súbor `standalone.xml` je hlavný konfiguračný súbor aplikovaného servera Wildfly a nachádza sa v adresári `/standalone/configuration`. SmartWeb server má implementovanú funkciu automatickej konfigurácie pri deploymente aplikácie aplikovaným serverom Wildfly. Z tohto dôvodu pre beh SmartWeb aplikácie ako takej nie je potrebné súbor `standalone.xml` editovať. Editácia je nevyhnutná iba v prípade úpravy konfigurácie samotného aplikovaného servera - napr. zmena portov, konfigurácia zabezpečenia komunikácie HTTPS a HTTP hlavičiek, zapnutie overovania cez klientské certifikáty, kompresia komunikácie at.



Pozor, editáciu konfiguračného súboru `standalone.xml` vykonávame zásadne pri vypnutom aplikovanom serveri, z dôvodu že počas jeho behu si ju on sám spravuje a môže prepísať zmeny uložené cez editor.

Odporúčaná konfigurácia pre optimálny beh a zabezpečenie servera

Pre optimálny beh SmartWeb servera a základné zabezpečenie cez HTTP hlavičky odporúčame nasledovnú konfiguráciu:

Odporúčané zmeny v standalone.xml

```
<?xml version='1.0' encoding='UTF-8'?>
<server xmlns="urn:jboss:domain:4.2">
  ...
  <profile>
    <subsystem xmlns="urn:jboss:domain:logging:3.0">
      ...
      <!-- Vypnutie zbytočných info hlášok o ukončení websocket spojenia -->
      <logger category="org.cometd.websocket.server.WebSocketTransport$WebSocketScheduler$1">
        <level name="WARN"/>
      </logger>
      ...
    </subsystem>
    ...
    <subsystem xmlns="urn:jboss:domain:undertow:3.1">
      <server name="default-server">
        <host name="default-host" alias="localhost">
          <location name="/" handler="welcome-content"/>
          <filter-ref name="gzipFilter" predicate="not min-content-size(450)"/>
          <filter-ref name="Strict-Transport-Security-header"/>
          <filter-ref name="Vary-header"/>
          <filter-ref name="X-Frame-Options"/>
          <filter-ref name="X-Content-Type-Options"/>
          <filter-ref name="X-XSS-Protection"/>
          <filter-ref name="Referrer-Policy"/>
          <filter-ref name="Content-Security-Policy"/>
        </host>
      </server>
      ...
      <filters>
        <response-header name="Vary-header" header-name="Vary" header-value="Accept-Encoding"/>
        <response-header name="Strict-Transport-Security-header" header-name="Strict-Transport-Security" header-value="max-age=31536000; includeSubDomains"/>
        <!-- Nastavenia pre Cross-Origin Resource Sharing (nepoužívané/zakázané) -->
        <response-header name="Access-Control-Allow-Origin" header-name="Access-Control-Allow-Origin" header-value="*" />
        <response-header name="Access-Control-Allow-Methods" header-name="Access-Control-Allow-Methods" header-value="GET, POST, OPTIONS, PUT"/>
        <response-header name="Access-Control-Allow-Headers" header-name="Access-Control-Allow-Headers" />
      </filters>
    </subsystem>
  </profile>
</server>
```

```

header-value="accept, authorization, content-type, x-requested-with"/>
    <response-header name="Access-Control-Allow-Credentials" header-name="Access-Control-Allow-
Credentials" header-value="true"/>
    <response-header name="Access-Control-Max-Age" header-name="Access-Control-Max-Age" header-
value="1"/>
    <!-- Zakázané vkladanie stránok do frame (starší spôsob) -->
    <response-header name="X-Frame-Options" header-name="X-Frame-Options" header-value="DENY"/>
    <!-- Vynútené použitie MIME typu nastaveného v HTTP hlavičke -->
    <response-header name="X-Content-Type-Options" header-name="X-Content-Type-Options" header-
value="nosniff"/>
    <!-- Zakázané zobrazenie stránky, ak bol detekovaný cross-site scripting (XSS) útok -->
    <response-header name="X-XSS-Protection" header-name="X-XSS-Protection" header-value="1;
mode=block"/>
    <!-- Neodosielanie referrer informácií -->
    <response-header name="Referrer-Policy" header-name="Referrer-Policy" header-value="no-referrer"
/>

    <!-- Nastavenie bezpečnostnej politiky obsahu:
    - zakázané vkladanie do frame (nový spôsob)
    - predvolene povolený zdroj obsahu z hostiteľskej domény
    - pre CSS štýly povolené aj zabezpečené https odkazy a inline
    - pre súbory písom povolené aj google písma
    - pre skripty povolené inline aj evaluácia
    - zakázané plugin objekty (flash a pod.)
    - povolené pripájanie z ubovolnej lokality
    -->
    <response-header name="Content-Security-Policy" header-name="Content-Security-Policy" header-
value="frame-ancestors 'none'; default-src 'self'; style-src https: 'self' 'unsafe-inline'; font-src 'self'
https://themes.googleusercontent.com https://fonts.gstatic.com; script-src 'self' 'unsafe-inline' 'unsafe-
eval'; object-src 'none'; connect-src */>
    <gzip name="gzipFilter"/>
    </filters>
    </subsystem>
    </profile>
    ...
    <interfaces>
    ...
    <interface name="public">
        <!-- Nastavenie bind adresy na všetky sieťové interface, kvôli tomu aby bol Wildfly prístupný aj z
vonku -->
        <inet-address value="${jboss.bind.address:0.0.0.0}"/>
    </interface>
    </interfaces>

    <socket-binding-group name="standard-sockets" default-interface="public" port-offset="${jboss.socket.
binding.port-offset:0}">
    ...
    <!-- Nastavenie portov pre HTTP a HTTPS na všeobecne používané hodnoty, pozor toto nemeni pri
inštalácii na Linuxe, vi kapitola o inštalácii na Linuxe nižšie -->
    <socket-binding name="http" port="${jboss.http.port:80}"/>
    <socket-binding name="https" port="${jboss.https.port:443}"/>
    ...
    </socket-binding-group>
</server>

```

Voliteľná konfigurácia silného šifrovania pre HTTPS

Nasledujúce zmeny v standalone.xml konfigurujú zapnutie silných šifrov pre HTTPS protokol. Podmienkou je inštalácia Java Cryptography Extensions popísaná v kapitole [Inštalácia JRE 1.8 a Git klienta](#).

Zmeny v standalone.xml pre HTTPS

```
<?xml version='1.0' encoding='UTF-8'?>
<server xmlns="urn:jboss:domain:4.2">
  ...
  <system-properties>
    <!-- Minimálna dĺžka Diffie-Helman kúba -->
    <property name="jdk.tls.ephemeralDHKeySize" value="2048"/>
  </system-properties>

  <management>
  <security-realms>
    ...
    <!-- Security realm undertowTLSRealm je potrebné nastavi iba v prípade konfigurácie HTTPS, zároveň
aj pre overovanie cez klientské certifikáty -->
    <security-realm name="undertowTLSRealm">
      <server-identities>
        <ssl protocol="TLS">
          <!-- Cesta/heslo ku keystore kde je uložený SSL certifikát pre HTTPS s definovaným
aliasom, napr. nblmgrel.ipesoft-int.sk -->
          <keystore path="server.jks" relative-to="jboss.server.config.dir" keystore-password="
secret" alias="nblmgrel.ipesoft-int.sk" key-password="secret"/>
        </ssl>
      </server-identities>
      <!-- as "authentication" je potrebné nastavi iba pre overovanie cez klientské certifikáty -->
      <authentication>
        <truststore path="client-certificates.jks" relative-to="jboss.server.config.dir" keystore-
password="secret"/>
      </authentication>
    </security-realm>
  </security-realms>
  ...
</management>

  <profile>
    ...
    <subsystem xmlns="urn:jboss:domain:undertow:3.1">
      ...
      <server name="default-server">
        ...
        <!-- !!! Atribút verify-client je potrebné nastavi na hodnotu REQUESTED v prípade ak je
potrebná autentifikácia cez klientské certifikáty, inak ho netreba meniť -->
        <https-listener name="https" socket-binding="https" security-realm="undertowTLSRealm" verify-
client="NOT_REQUESTED"
          enabled-protocols="TLSv1.2,TLSv1.1"
          enabled-cipher-suites="TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA,
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA,TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256,
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384,TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA,
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,
TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA,TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA,TLS_DHE_RSA_WITH_AES_128_GCM_SHA256,
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384,TLS_DHE_RSA_WITH_AES_128_CBC_SHA,TLS_DHE_RSA_WITH_AES_256_CBC_SHA,
TLS_DHE_RSA_WITH_AES_128_CBC_SHA256,TLS_DHE_RSA_WITH_AES_256_CBC_SHA256"
          enable-http2="true"/>
      </server>
    </subsystem>
  </profile>
</server>
```

Uvedené zmeny konfigurácie majú nastavený zoznam povolených protokolov a šifier tak, aby bol bezpečný, podporené sú len novšie prehliadače - IE11 a Android >= 4.4.3, Safari >= 7. V prípade potreby podpory ešte starších prehliadačov treba nastaviť inak tieto atribúty:

```

enabled-protocols="TLSv1.2,TLSv1.1,TLSv1"
enabled-cipher-suites="TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA,TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA,TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256,
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384,TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA,
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA,
TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA,TLS_DHE_RSA_WITH_AES_128_GCM_SHA256,TLS_DHE_RSA_WITH_AES_256_GCM_SHA384,
TLS_DHE_RSA_WITH_AES_128_CBC_SHA,TLS_DHE_RSA_WITH_AES_256_CBC_SHA,TLS_DHE_RSA_WITH_AES_128_CBC_SHA256,
TLS_DHE_RSA_WITH_AES_256_CBC_SHA384,TLS_DHE_DSS_WITH_AES_128_GCM_SHA256,TLS_DHE_DSS_WITH_AES_256_GCM_SHA384,
TLS_DHE_DSS_WITH_AES_128_CBC_SHA256,TLS_DHE_DSS_WITH_AES_256_CBC_SHA256,TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256,
TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384,TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256,
TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384,TLS_RSA_WITH_AES_128_GCM_SHA256,TLS_RSA_WITH_AES_256_GCM_SHA384,
TLS_RSA_WITH_AES_128_CBC_SHA256,TLS_RSA_WITH_AES_256_CBC_SHA256,TLS_RSA_WITH_AES_128_CBC_SHA,
TLS_RSA_WITH_AES_256_CBC_SHA"

```

Volitená konfigurácia automatického presmerovania HTTP na HTTPS

V prípade že Wildfly AS je dostupný z vonku priamo cez ním otvorené porty a doménu je potrebná nasledovná konfigurácia:

```

<?xml version='1.0' encoding='UTF-8'?>
<server xmlns="urn:jboss:domain:4.2">
    ...
    <profile>
    ...
    <subsystem xmlns="urn:jboss:domain:undertow:3.1">
        ...
        <server name="default-server">
            <!-- Atribút redirect-socket nastavi iba v prípade výhradnej komunikácie cez
HTTPS -->
            <http-listener name="default" socket-binding="http" redirect-socket="https"/>
            </server>
        ...
    </subsystem>
    </profile>
</server>

```

V prípade že Wildfly AS je dostupný cez samostatný proxy server alebo IPTABLES rerouting je potrebná nasledovná konfigurácia:

```

<?xml version='1.0' encoding='UTF-8'?>
<server xmlns="urn:jboss:domain:4.2">
    ...
    <profile>
    ...
    <subsystem xmlns="urn:jboss:domain:undertow:3.1">
        ...
        <server name="default-server">
            ...
            <host name="default-host" alias="localhost">
                ...
                <!-- Atribút predicate treba nastavi na HTTP port definovaný v
poslednej asti standalone.xml-->
                <filter-ref name="http-to-https" predicate="equals(%p,8080)"/>
                ...
            </host>
            ...
            <filter-ref name="http-to-https" predicate="equals(%p,8080)"/>
            </server>
            ...
            <filters>
                <!-- Atribút target treba nastavi na finálnu doménu a port, %U je placeholder
pre zvyšnú as otvárajúcu url linku-->
                <rewrite name="http-to-https" redirect="true" target="https://myhostname:8443%U"
/>
            </filters>
        </subsystem>
    </profile>
</server>

```