V.B.G.

Virtual Business Gate

Vedi note: documenti / contributi degli enti riusatori / README.md

SOMMARIO

1 Introduzione

Lo scopo del documento è quello di raccogliere tutte le informazioni necessarie per configurare l'ambiente di sviluppo della piattaforma VBG (backoffice e frontoffice). Il documento fornisce le indicazioni per configurare l'ambiente di sviluppo nel modo corretto e secondo gli standard decisi.

Verranno elencati, spiegati e configurati :

- Strumenti di sviluppo (IDE , plugin, client sql, etc..)
- Frameworks utilizzati
- Metodologie di sviluppo
- Processi

2 Strumenti di sviluppo e configurazione componenti java

Lista strumenti principali di sviluppo:

- IDE : Eclipse
- Linguaggio : JAVA
- Servlet Container : Tomcat
- Plugins principali : Subeclipse, Hibernate tools, ResourceBundle Editor, Mylyn, redmine

2.1. IDE

L' IDE utilizzato per lo sviluppo è Eclipse versione Luna; Creare sul proprio PC la cartella "SVILUPPO" (Es: C://SVILUPPO), andare a copiare e spacchettare eclipse all'interno della cartella creata e rinominare la cartella di installazione come **eclipse_luna** C://SVILUPPO/eclipse_luna.

Configurazioni base

Passo 1:
All'interno della cartella ./SVILUPPO creare le sotto cartelle
workspaces/workspace_<progetto>

Passo 2:
Aprire eclipse e selezionare il workspace creato al passo precedente e procedere alla configurazione base.

1. *Deve essere impostato tutto il workspace con la codifica UTF-8, andare in windows→preferences e cercare:*

- **jsp file** : mettere la codifica UTF-8
- **css files** : mettere la codifica UTF-8
- **html Files** : mettere la codifica UTF-8
- **workspace** : mettere la codifica UTF-8

2. *Impostare il formatter standard di sviluppo:*

- Andare nella cartella \\fileserver2\E$\template standard sviluppo
- Copiare il file *formatter.xml*
- In eclipse andare in *windows->preference* e cercare ***Java-code style-formatter*** ed importare il file appena salvato. Se l'operazione è corretta dovrebbe comparire la dicitura "Java convention + 150 line width".

3. *Impostare riga template riga di debug:*

Andare su *Windows --> preferences --> java --> editor --> templates* , cliccare su "New" e popolare i campi con i dati seguenti.

- Name : deguglog
- Contex: java
- Description : riga di debug
- Patter:

```
if(log.isDebugEnabled()){
    log.debug("${enclosing_method}# ${cursor}");
}
```

Se configurato all'interno di ogni classe java scrivendo debuglog + ctrl + barra spaziatrice verrà riportato il template salvato per il log

2.2. **JAVA**

All'interno della macchina deve essere installa la JDK versione 6.45. Scegliere quella corretta per la propria macchina:

1. Tipo sistema operativo
2. 64 bit o 32 bit

Copiare il file sul proprio pc e procedere con l'installazione.

2.3. **Tomcat**

Installare nella macchina il Servlet Container **Tomcat**; la versione attuale da utilizzare è Tomcat 6.43 anche in questo caso dovrà essere scelto quello corretto per le caratteristiche della macchina che si usa:

1. Tipo sistema operativo
2. 64 bit o 32 bit

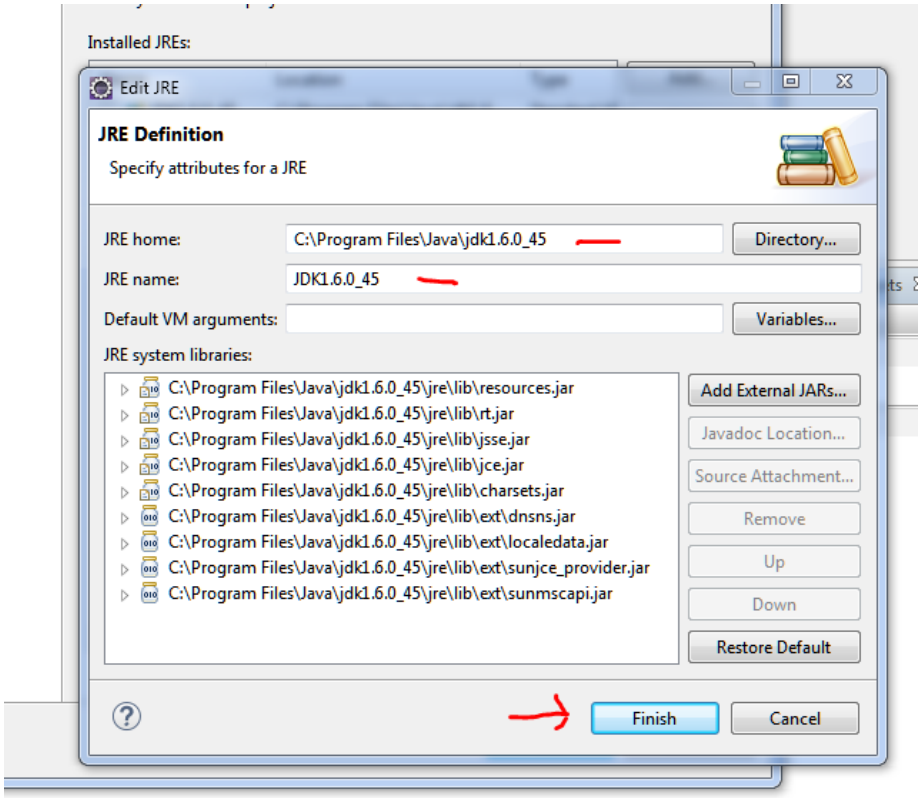
Copiare i file di installazione sul proprio PC e cominciare la procedura di installazione

A. Creare una nuova sotto cartella all'interno della cartella SVILUPPO con nome ***tomcat***
Es C://SVILUPPO/tomcat
B. Scompattare il pacchetto di installazione all'interno di questa cartella, la nuova cartella creata sarà del tipo apache-tomcat-6.0.xx

2.4. **Configurazione di JAVA e TOMCAT in Eclipse**

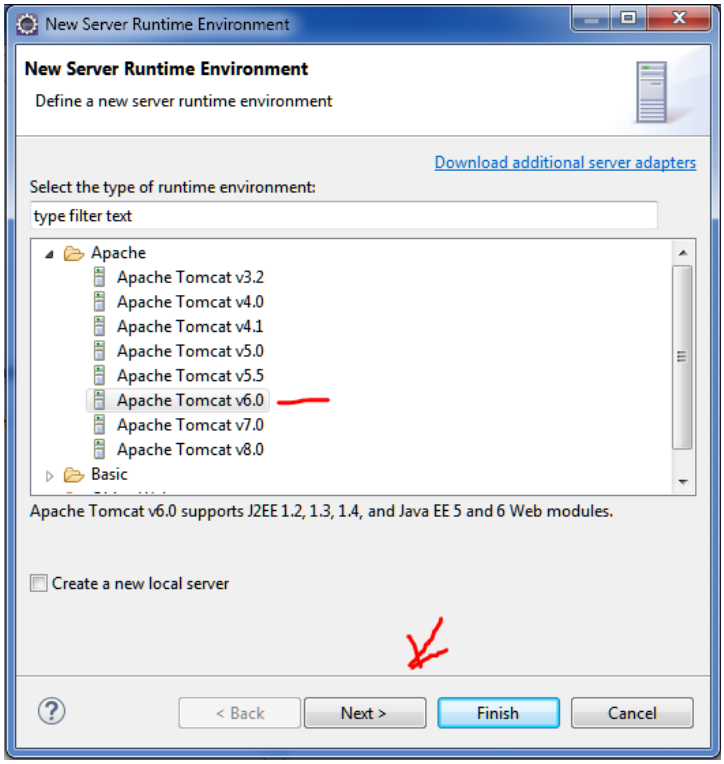
Finite le procedure di installazione di JAVA e TOMCAT le andremo a configurare all'interno di Eclipse;
apriamo il programma e scegliamo il workspace creato.
Andare in *windows→preferences* e selezionare **Installed JRES**; sulla colonna di destra cliccare su "Add", si aprirà un pannello, dovremo impostare la

- JRE Home : ../jdk1.6.0_45 (mettere il percorso dove si trovata la jdk installata)
- JRE Name : JDK1.0_45



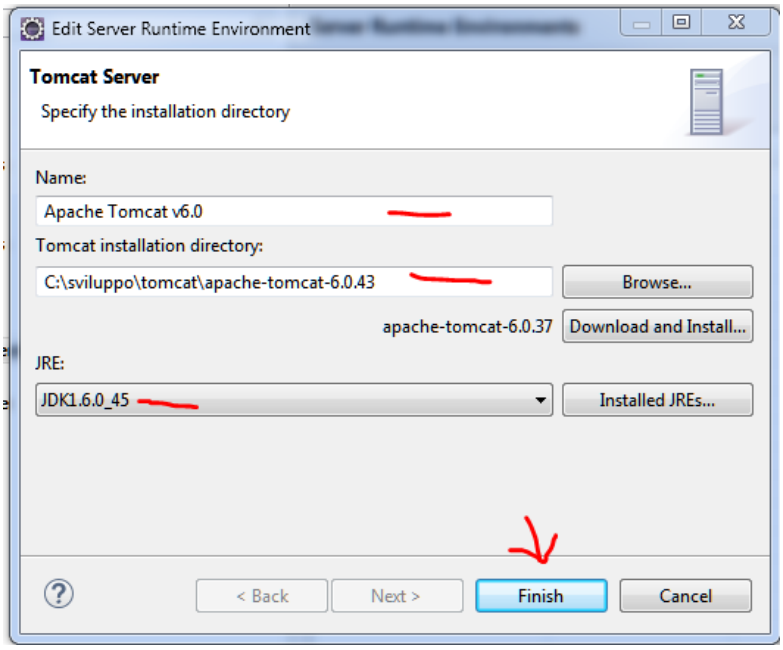
Cliccare su “Finish”

Andare in *windows→preferences* e selezionare **Runtime Enviroments;** sulla colonna di destra cliccare sul bottone “Add” nella schermata che si apre selezioniamo Apache Tomcat 6



Poi clicchiamo su “Next”; nel pannello successivo andremo a popolare

a. Name
b. Directory installazione tomcat
c. JRE la JDK configurata nel passaggio precedente



Finiti questi passaggi l'ambiente è pronto per poter scaricare il progetto e fare il deploy in sviluppo.

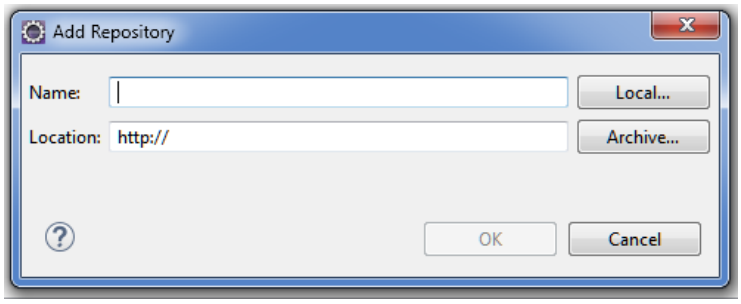
2.5. **Installazione e configurazione plugins principali per lo sviluppo**

Lista plugins principali:

- Subeclipse
- Hibernate tools
- Resource bundle Editor
- Mylyn
- Connettore per redmine

2.1.1 2.5.1 Installazione

Andare in *Help* →*Install new software* e cliccare su "Add"

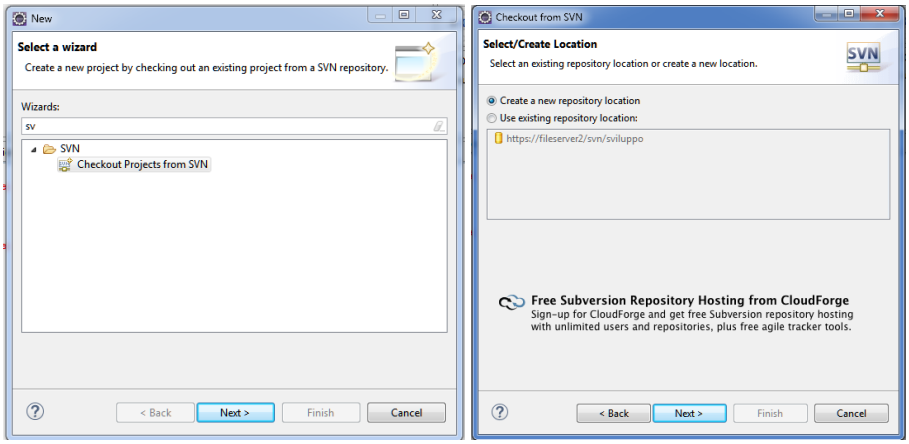


Inserire il nome : Es. subeclipse
Location : L' indirizzo dove recuperare la plugin
Cliccare su ok e poi iniziare la procedura di installazione

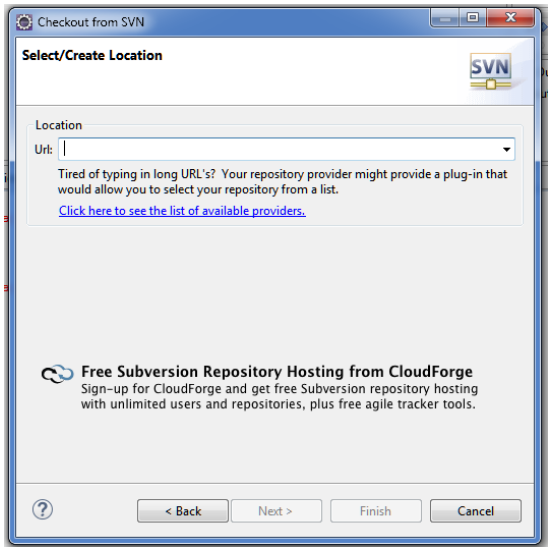
- Subeclipse: http://subclipse.tigris.org/update_1.10.x
- Hibernate tools: http://download.jboss.org/jbosstools/updates/stable/luna/
- Resource bundle Editor: https://raw.githubusercontent.com/essiembre/eclipse-rbe/master/eclipse-rbe-update-site/site.xml

2.1.2 2.5.2 Configuazione

Configurazione subeclipse: aprire eclipse e andare su *File →New →Others*



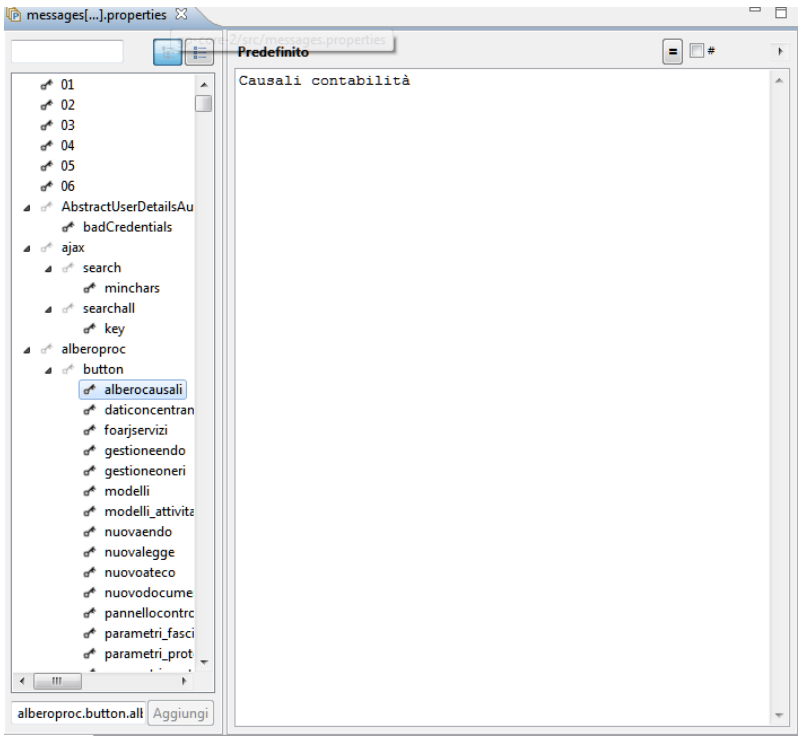
Cliccare su next, poi selezionare "Create new repository location" e cliccare ancora su next



Inserire l'url : cliccare su next; ci verrà chiesta un username e password,
andremo ad utilizzare quelle di dominio (quelle per accedere al PC). Se l'operazione è andata a buon fine verrà mostrata tutta la lista dei progetti presenti sul *svn/sviluppo*

Configurazione Hibernate tools : per la configurazione ed utilizzo del pluging si rimanda la documento *"configurazione hibernate tool.doc"*

Configurazione ResourceBundleEditor: se installato correttamente un file di tipo *.properties* se aperto con questo plugin avrà questa forma



Cliccando su una voce posta sulla colonna di destra sarà possibile vedere per quella chiave che valore corrisponde.

Nel caso dobbiamo inserire una nuova voce, andiamo a scrivere la key nella text box vicino al bottone aggiungi, inserendo la key (Es label.testo.campo1) potranno accadere due cose:

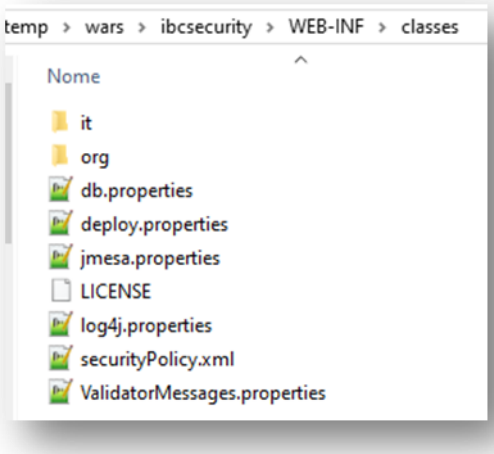
1. La label non esiste si attiva il campo “Aggiungi” e facendo click potremmo inserire il testo da associare alla label all’interno della text area sulla destra
2. La label esiste il campo “Aggiungi” rimane disabilitato e viene mostrato il testo all’interno della text area sulla destra ed eventualmente possiamo modificarlo se necessario.

2.2 configurazione dei progetti in eclipse

La piattaforma VBG è composta da una serie di applicativi scaricabili con pacchetti war.
Procediamo alla configurazione di uno war di esempio

- ibcsecurity.war

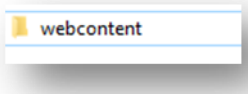
Passo 1: Scompattazione dello war
Recuperare lo war ibcsecurity.war e scompattarlo in una directory temporanea.
Posizionarsi nella directory WEB-INF/classes



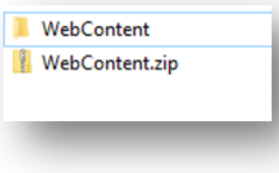
ricercare tutti i file .class e rimuoverli

Spostare la cartella classes e posizionarla in una cartella esterna rinominandola in src

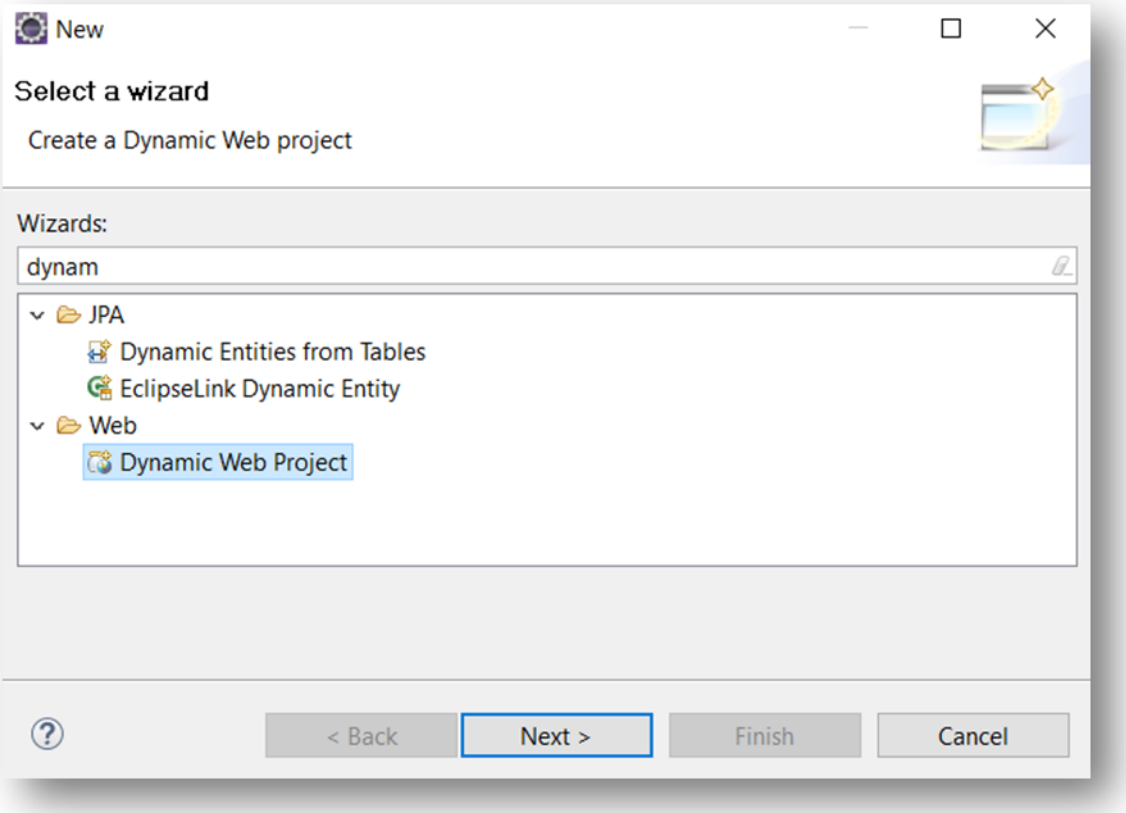
Creare una cartella WebContent e muovere il contenuto della directory ibcsecurity dentro

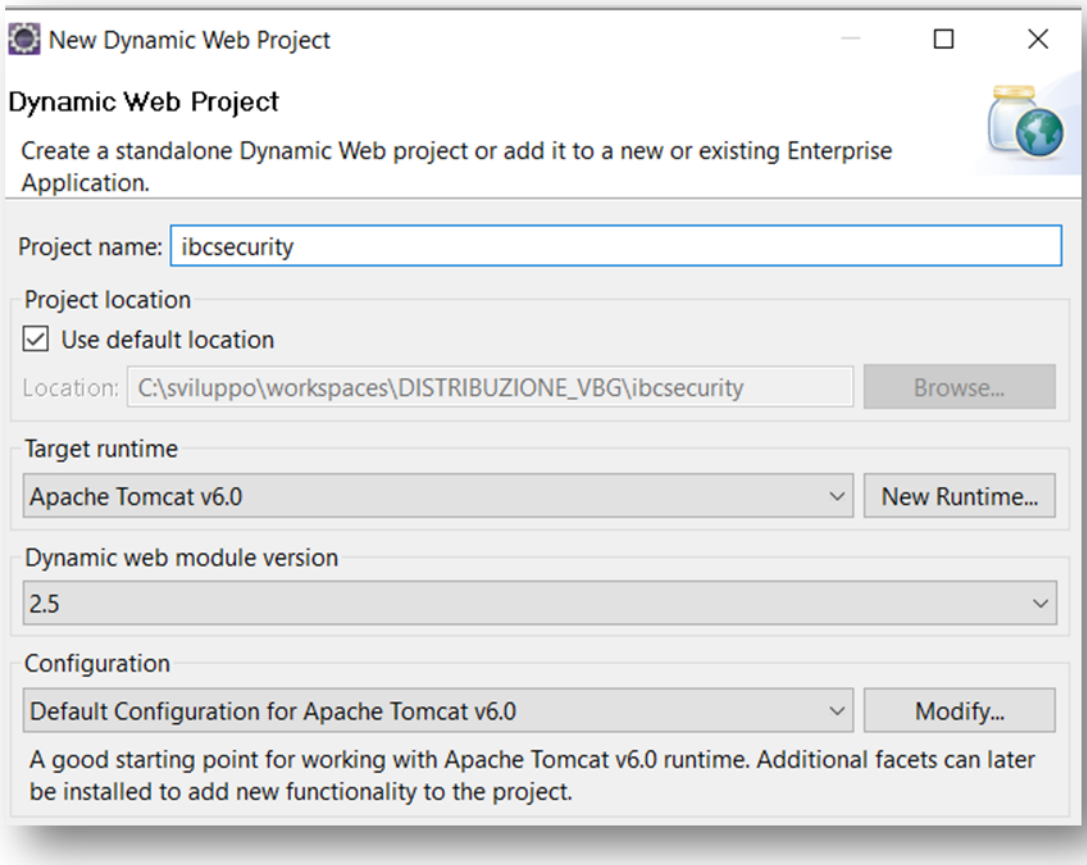


Zippare il contenuto della cartella WebContent

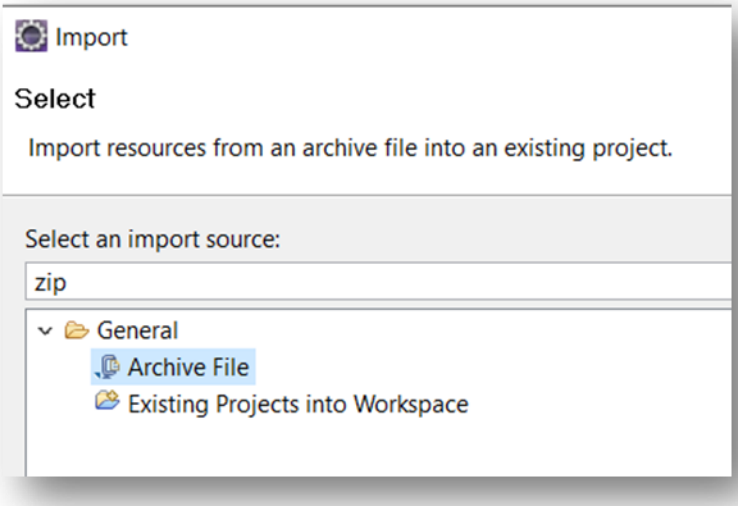


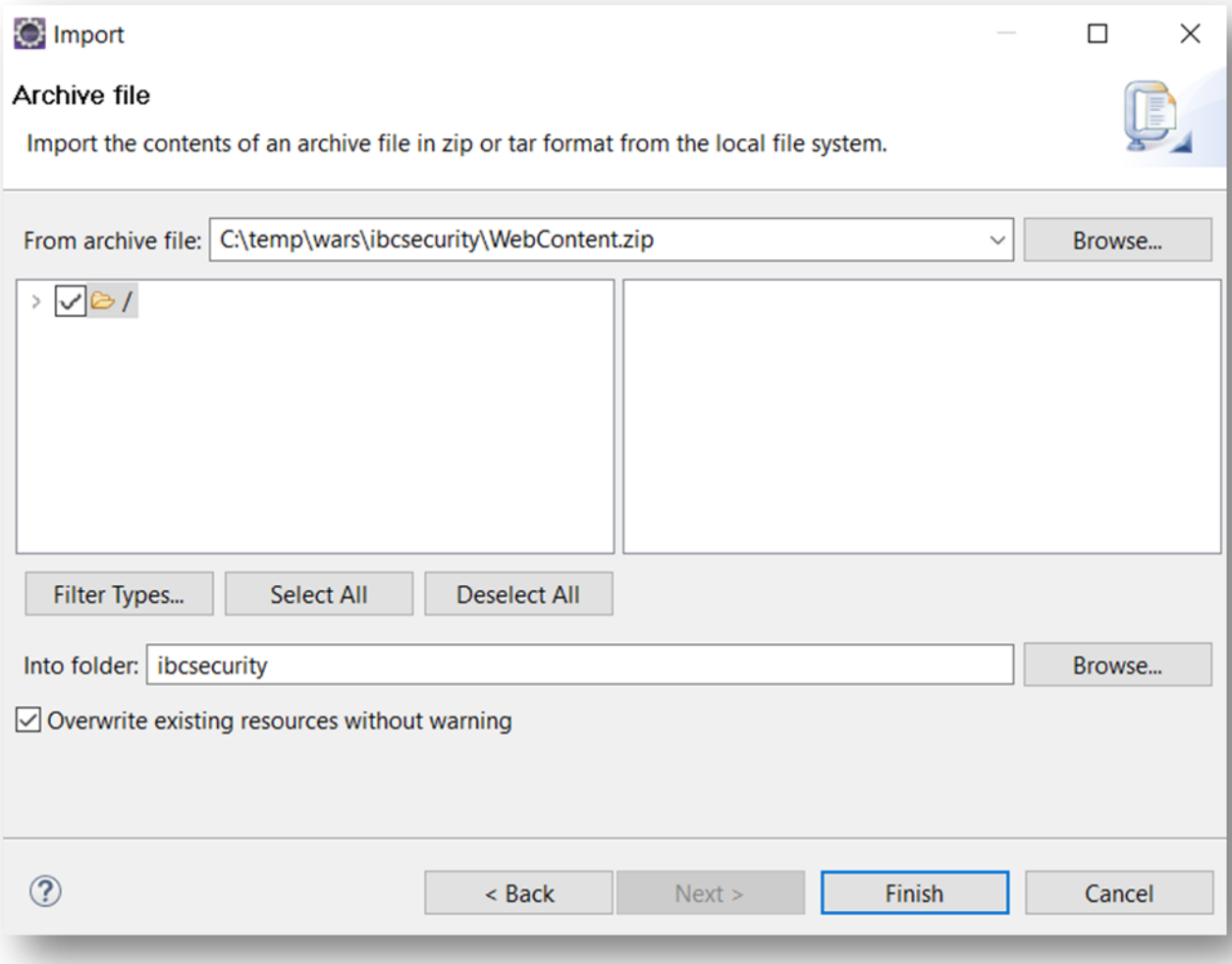
Creare nello workspace un nuovo progetto ibcsecurity



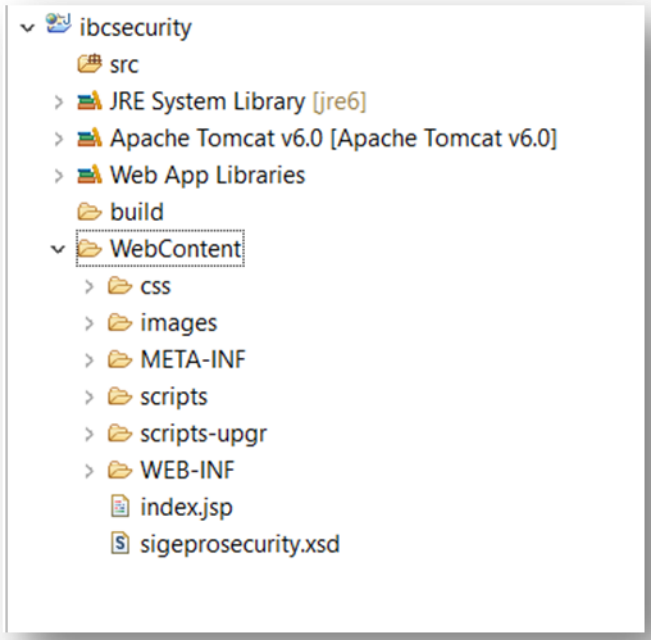


Importare lo zip WebContent nel progetto

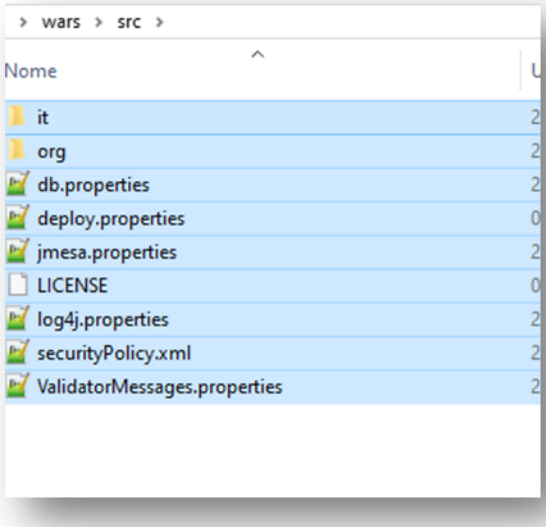




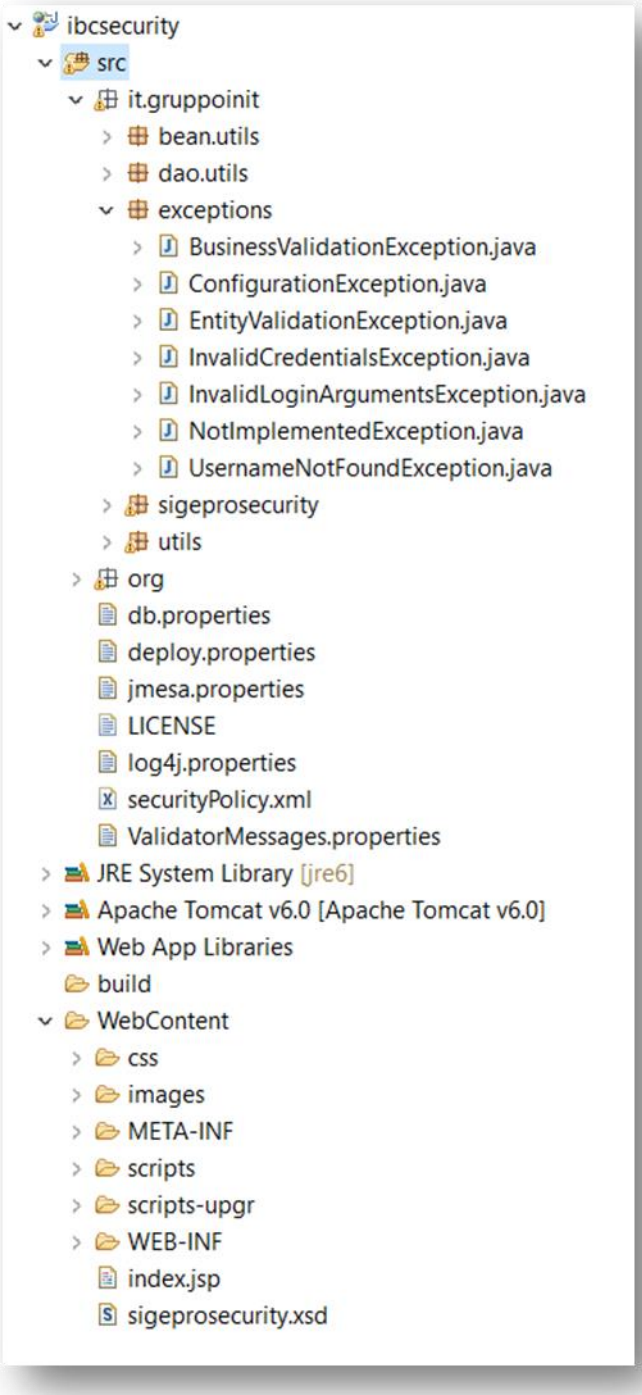
Dovrebbe comparire il layout di progetto che segue



Copiare il contenuto della cartella src preso prima

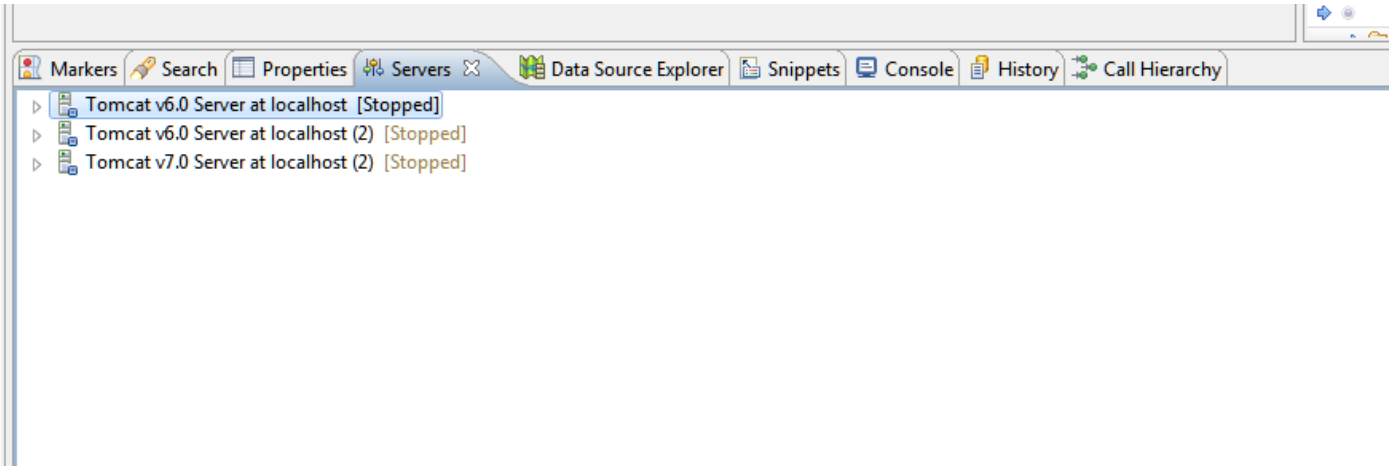


Ed incollarlo nella directory src del progetto

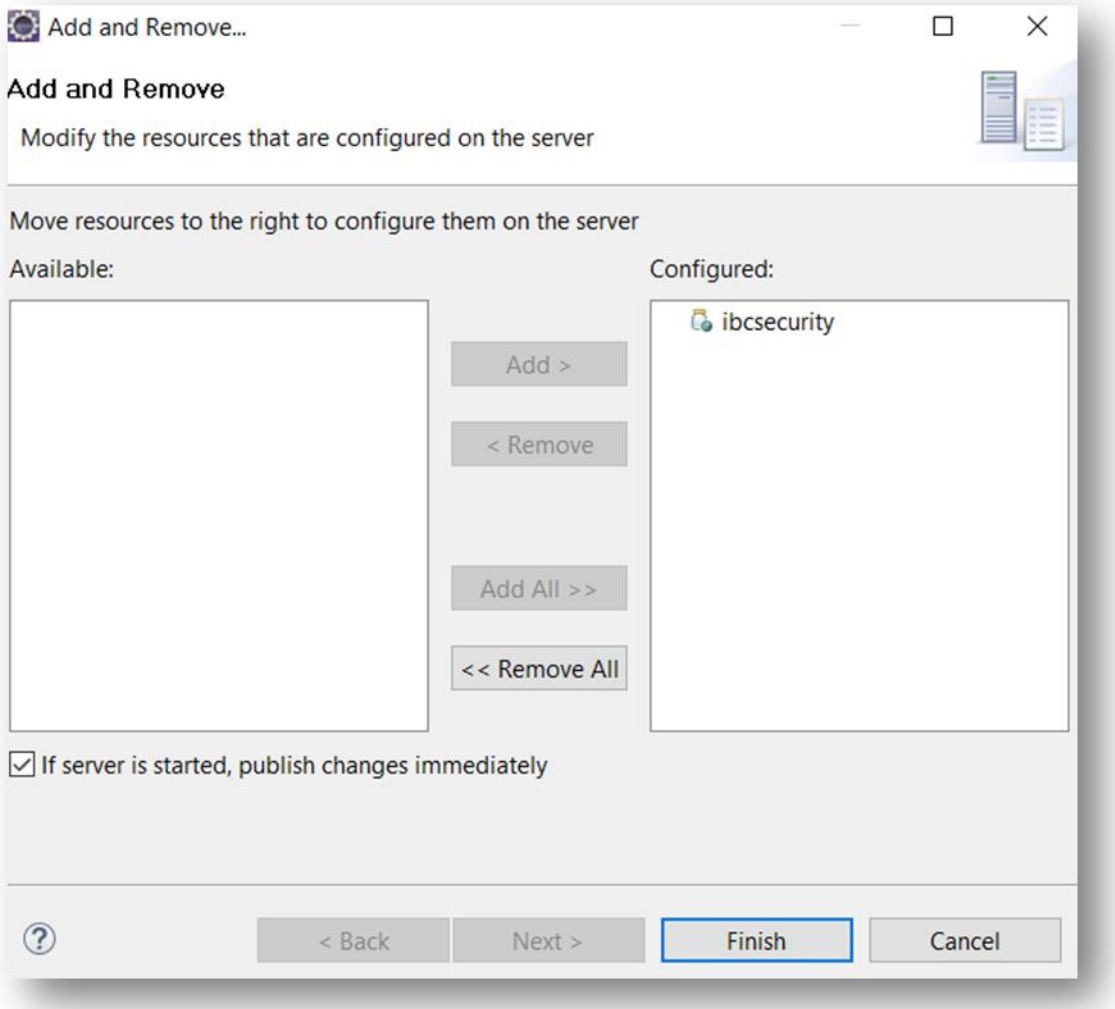


2.3 Deploy dei progetti sul Tomcat di sviluppo

Una volta finita la procedura di dowload del progetto siamo pronti per fare il deploy (L'esempio riguarderà il deploy del backend). Tornare nella schermata principale di Eclispe e aprire la prospective view "Server"

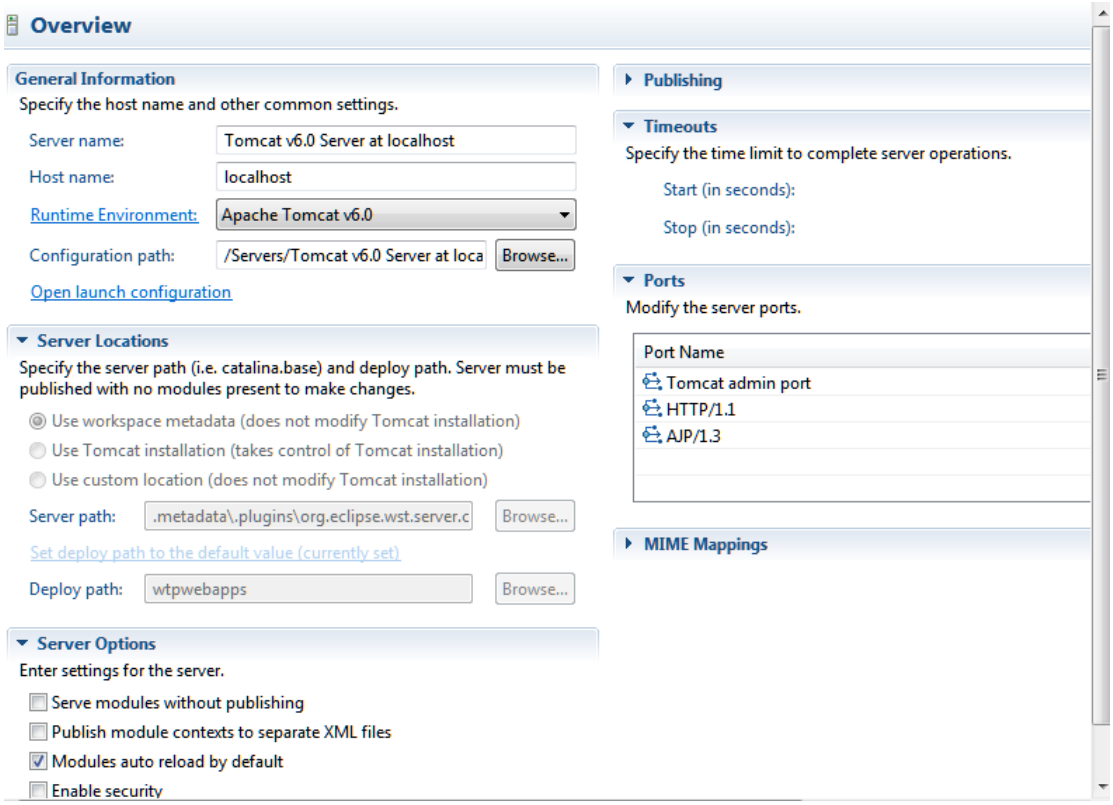


Se la procedura di installazione e configurazione del Tomcat è stata fatta correttamente avremo uno o più istanze del server presenti.
Selezionare il server, tasto destro e cliccare su *"Add and Remove"* e aggiungere il progetto ibcsecuritye cliccare su *"Finish"*

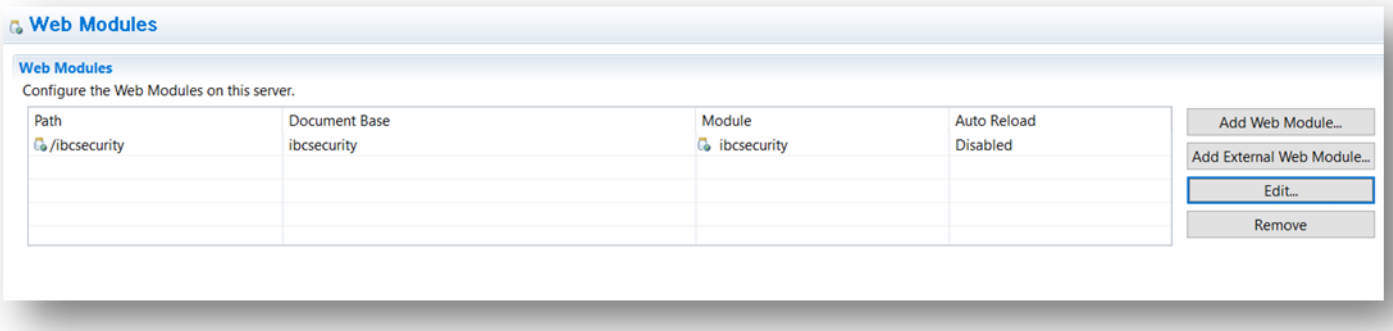


Tornati sulla view “Server” fare doppio click sul server configurato; nella schemata “Overview”, settare il tempo di:

- Start : 300
- End : 45



Nella schermata di “Modules” disabilitare “Auto reload ” per il progetto



Questo ci permetterà in fase di sviluppo di non dovere riavviare il Tomcat se cambiamo righe di codice. Il riavvio sarà necessario solo se andremo ad aggiungere una Classe, Un metodo o cambieremo la segnatura a un metodo esistente.
Settare i parametri di memoria per il deploy del progetto. Andare su Run→Debug Configuration

In *VM Argument* settare i seguenti parametri:

```
-Xms1024m

-Xmx2048m

-XX:PermSize=256m

-XX:MaxPermSize=256m
```

Prima di mandare in esecuzione il Tomcat, l'ultimo passaggio sarà quello verificare le configurazioni di connessione del progetto.
Aprire il file *deploy.properties* di ibcsecurity

```
##------- DRIVER ----------------------------------------------##
#
# com.mysql.jdbc.Driver
# oracle.jdbc.OracleDriver
# org.postgresql.Driver
# net.sourceforge.jtds.jdbc.Driver
##-----------------------------------------------------------##

jdbc.driverClassName = oracle.jdbc.OracleDriver
##-------STRINGHE DI CONNESSIONE-----------------------------##
#
# jdbc:oracle:thin:@HOST:1521:SID
# jdbc:postgresql://HOST:5432:/DATABASE
# jdbc:mysql://HOST:3306/DATABASE?autoReconnect=true
# jdbc:jtds:sqlserver://DBMSSQL2005:1433/sigepro
#
##-----------------------------------------------------------##

jdbc.url = jdbc:oracle:thin:@dbora10g:1521:ora10g
jdbc.username =
jdbc.password =
##-----------------------------------------------------------##
##--------------PARAMETRI PER LA CONFIGURAZIONE FUNZIONE ELIMINA TOKEN---------##
saveTextFile=true
## schedulazione
##false=disabilitata, true=abilitata
scheduler.enable=true
##tipo cron, simple
scheduler.type=cron
#espressione per tipo cron (Seconds,Minutes,Hours,Day of month,Month,Day of week,Year)
scheduler.cron.expression=0 30 1 ? * *
#valori per tipo simple (millis)
```

Finita tutta questa fase di configurazione possiamo avviare il Tomcat

2.4 Lista delle componenti java

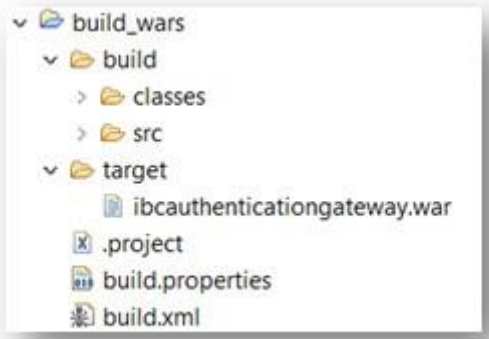
Di seguito la lista delle componenti di progetto e la funzionalità che assolvono

Componente	Tipologia	Funzionalità
EGrammataWsProxy.war	MiddleWare	Plugin per integrazione con protocollo EGrammata
FederaAG.war	Web app autenticazione	Plugin per autenticazione con Sistema FedERa
LoginUmbriaAG.war	Web app autenticazione	Plugin per autenticazione con Sistema LoginUmbria
MailService.war	Middelware	Web app per invio mail
NLAPeople.war	Plugin ricezione pratiche da sistemi PEOPLE SUAPER	
areariservata2.war	Web app Middleware	Servizi rest CMS frontoffice
backend.war	Webapp utenti backoffice	Gestionale di Backoffice
console.war	Webapp utenti	Cms di Backoffice per i servizi di CONSOLE

	backoffice	
download-app.war	Middleware	Plugin per scaricare da front documenti salvati nel documentale
dss-webapp.war	Middleware	Webapp verifica file firmati digitalmente
firma.war	Middleware	Webapp per firma file mediante interfaccia web o web service
ibcauthenticationgateway.war	Webapp autenticazione	Componente di autenticazione frontoffice e backoffice
ibcfileconverter.war	Middleware	Componente di conversione dei file in vari formati
ibcsecurity.war	Middleware	Componente per la sicurezza dei token di autenticazione
ibcstc.war	Middleware	Componente di orchestrazione e cooperazione applicativa tra vari sistemi basato su modellazione di progetto
nla-enti.war	Middleware	Plugin per invio messaggi standard DPR 160
nla-pdd-ri.war	Middleware	Plugin per comunicazione con impresainungiorno
nla-pec.war	Middleware	Componente per la lettura dei messaggi di caselle PEC
pdfutils.war	Middleware	Componente per lettura scrittura su file PDF
servizi-console.war	Middleware	Webapp per servizi json dati CMS della console
stc-mobile-services.war	Middleware	Componente per servizi json su pratiche
vbg-payer-adapter.war	Middleware	Componente di integrazione con sistema PayER
wsanagrafe2.war	Middleware	Componente di integrazione con sistemi legacy degli enti
wssit.war	Middleware	Componente di integrazione con sistemi legacy dell'ente

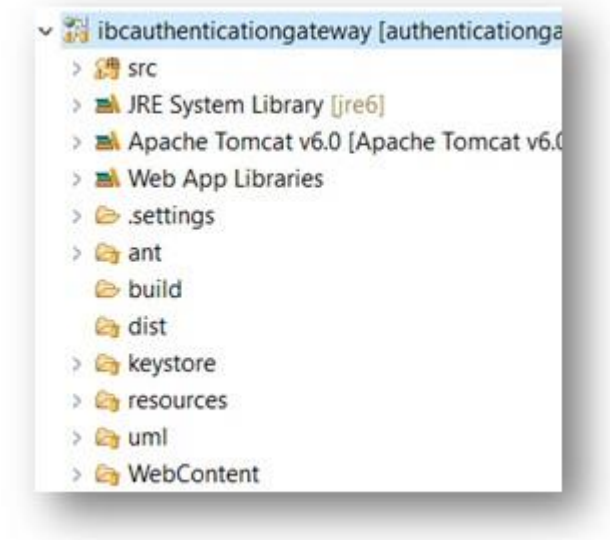
2.5 Configurazioni di build mediante script ant

È possibile compilare i progetti anche mediante script ANT.



Per ogni progetto va specificato un build.properties specifico dove vanno indicati il nome del progetto (sarà anche il nome dello war)
e la directory base (root dell'applicativo) installato secondo la struttura

ibcauthenticationgateway
/src
/WebContent



Contenuto del file
build.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<project basedir="." default="war" name="WarBuilder">

      <property file="build.properties"/>

    <property environment="env"/>
    <property name="debuglevel" value="source,lines,vars"/>
    <property name="target" value="1.6"/>
    <property name="source" value="1.6"/>

    <path id="Apache Tomcat v6.0 [Apache Tomcat v6.0].libraryclasspath">
      <fileset dir="${tomcat.home}/lib/" includes="*.jar"/>
    </path>
    <path id="Web App Libraries.libraryclasspath">
      <fileset dir="${project.folder.base}/WebContent/WEB-INF/lib/"
includes="*.jar"/>
    </path>
    <path id="classpath">
        <pathelement location="build/classes"/>
        <path refid="Apache Tomcat v6.0 [Apache Tomcat
v6.0].libraryclasspath"/>
        <path refid="Web App Libraries.libraryclasspath"/>
    </path>
    <target name="init">
        <mkdir dir="build/classes"/>
      <mkdir dir="build/src"/>
      <mkdir dir="target"/>
        <copy includeemptydirs="false" todir="build/classes">
            <fileset dir="${project.folder.base}/src">
                <exclude name="**/*.launch"/>
                <exclude name="**/*.java"/>
            </fileset>
        </copy>
        <copy includeemptydirs="false" todir="build/src">
            <fileset dir="${project.folder.base}/src">
            </fileset>
```

```
        </copy>

    </target>
    <target name="clean">
        <delete dir="build/classes"/>
      <delete dir="build/src"/>
    </target>
    <target depends="clean" name="cleanall"/>
    <target depends="build-subprojects,build-project" name="build"/>
    <target name="build-subprojects"/>

      <target depends="init" name="build-project">
        <echo message="${ant.project.name}: ${ant.file}"/>
        <javac debug="true" debuglevel="${debuglevel}"
            destdir="build/classes" includeantruntime="false"
            source="${source}" target="${target}">
            <src path="build/src"/>
            <classpath refid="classpath"/>
        </javac>
    </target>

    <target description="Build all projects which reference this project.
Useful to propagate changes." name="build-refprojects"/>

      <target depends="cleanall,build" name="war">
            <echo message="creo il file war"/>
            <war destfile="target/${project.name}.war"
webxml="${project.folder.base}/WebContent/WEB-INF/web.xml">
              <fileset dir="${project.folder.base}/WebContent/"/>
              <classes dir="build/classes"/>
            </war>
      </target>


</project>
```

Contenuto del file build.properties

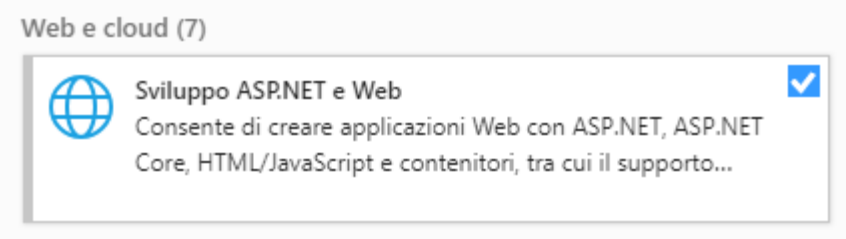
```
project.name = ibcauthenticationgateway

project.folder.base = ../ibcauthenticationgateway

tomcat.home=C:\\sviluppo\\apache-tomcat-6.0.43
```

3 Strumenti di sviluppo progetti .NET

Si utilizza Visual studio 2017 per la gestione / compilazione dei sorgenti. È necessario installare il modulo per il web development

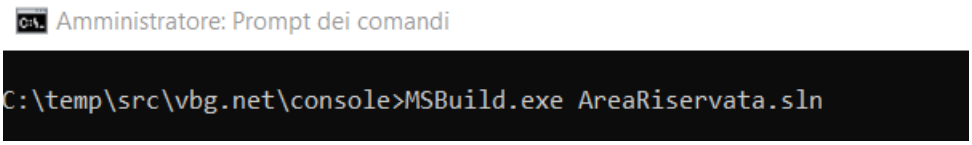


3.1 Informazioni di build

Per la compilazione si utilizza il normale tool dell'IDE.

È possibile compilare anche le soluzioni mediante comando da shell

- MSBuild.exe pathDellaSoluzione



3.2 Lista delle componenti .NET

Componente	Tipologia	Funzionalità
console/areariservata.sln	Webapp Frontend	Webapp presentazione domanda procedimenti comuni a tutti gli enti servizi di console
console/backoffice.sln	Middleware	Contiene una serie di servizi invocati dall'area riservata console
areariservata/areariservata.sln	Webapp Frontend	Webapp presentazione domanda procedimenti dello specifico ente
areariservata/backoffice.sln	Middleware	Contiene una serie di servizi invocati dall'area riservata

4 Strumenti di sviluppo portale informativo

Il portale informativo è un applicazione SPA (Single page application) scritta con Angular, html5.

4.1 IDE e configurazione ambiente di sviluppo

Si utilizza come IDE Visual Studio code

```
Versione: 1.39.2 (system setup)
Commit: 6ab598523be7a800d7f3eb4d92d7ab9a66069390
Data: 2019-10-15T15:35:18.241Z
Electron: 4.2.10
Chrome: 69.0.3497.128
Node.js: 10.11.0
V8: 6.9.427.31-electron.0
OS: Windows_NT x64 10.0.18362
```

Nel progetto sono incluse le info di build nei file che riportiamo di seguito

- Readme.txt

```
- PREREQUISITI
- ============
-
- Plugin di VSCode
- ----------------
- Si consiglia di utilizzare "Angular Extension Pack"
-
-
- Node.js
- -------
- Per poter compilare il progetto è necessario installare NOde.js all'ultima versione
  stabile, questo permette di avere npm disponibile nel sistema.
- I pacchetti npm globali utilizzati dal progetto sono:
- - Angular cli (https://cli.angular.io/): npm install -g @angular/cli
- - Less (http://lesscss.org/): npm install -g less
- - Less Watch Compiler (https://www.npmjs.com/package/less-watch-compiler): npm
  install -g less-watch-compiler
-
-
- Prima compilazione
- ------------------
```

- Dopo aver scaricato il progetto eseguire i seguenti comandi:
- - npm install
-
- Avviare il progetto in locale con
-
- npm run start
-
- Navigare con il browser all'indirizzo http://localhost:4200/
-
-
- Fare riferimento al file readme.md per le istruzioni di base per utilizzare la CLI.
-
- Opzionale: installare il plugin vsc-angular-cli per eseguire i comandi della CLI direttamente da Code
-
-
- Pubblicazione su IIS
- ----------
- Fare riferimento al file pubblicazione-su-iis.txt

- **Pubblicazione-su-iis-txt**

La guida è disponibile all'indirizzo:
https://blogs.msdn.microsoft.com/premier_developer/2017/06/14/tips-for-running-an-angular-app-in-iis/

Fondamentalmente occorre lanciare il comando

> ng build --prod

se si desidera pubblicare l'applicazione sulla root del web server, altrimenti utilizzare

> ng build --prod --base-href /nome-applicazione/

se si desidera pubblicare l'applicazione nel path http://www.miosito.com/nome-applicazione/

Una volta preparati i files di installazione occorre creare una virtual directory sotto IIS.

#Deep linking
Per permettere il deep linking ed evitare errori di tipo 404 quando si chiamano direttamente pagine dell'applicazione
senza passare per la home è necessario installare il package "url-rewrite" da
https://www.iis.net/downloads/microsoft/url-rewrite

Dopo aver configurato l'url-rewrite è necessario creare un nuovo web.config nella root

dell'applcazione che
permetta il redirect di url non esistenti. Un esempio di web.config è:

```
<configuration>
<system.webServer>
 <rewrite>
  <rules>
   <rule name="Angular Routes" stopProcessing="true">
    <match url=".*" />
        <conditions logicalGrouping="MatchAll">
         <add input="{REQUEST_FILENAME}" matchType="IsFile" negate="true" />
         <add input="{REQUEST_FILENAME}" matchType="IsDirectory" negate="true" />
        </conditions>
        <action type="Rewrite" url="/MyApp/" />
        <!--<action type="Rewrite" url="/" />-->
   </rule>
  </rules>
 </rewrite>
</system.webServer>
</configuration>
```

Per altre informazioni consultare la guida all'indirizzo

https://blogs.msdn.microsoft.com/premier_developer/2017/06/14/tips-for-running-an-angular-app-in-iis/

- **README.md**

```
# Ng2FrontofficeDesignItalia

This project was generated with [Angular CLI](https://github.com/angular/angular-cli) version 1.3.0.

## Development server

Run `ng serve` for a dev server. Navigate to `http://localhost:4200/`. The app will automatically reload if you change any of the source files.

## Code scaffolding

Run `ng generate component component-name` to generate a new component. You can also use `ng generate directive|pipe|service|class|guard|interface|enum|module`.
```

```
## Build

Run `ng build` to build the project. The build artifacts will be stored in the `dis
t/` directory. Use the `-prod` flag for a production build.

## Running unit tests

Run `ng test` to execute the unit tests via [Karma](https://karma-
runner.github.io).

## Running end-to-end tests

Run `ng e2e` to execute the end-to-
end tests via [Protractor](http://www.protractortest.org/).
Before running the tests make sure you are serving the app via `ng serve`.

## Further help

To get more help on the Angular CLI use `ng help` or go check out the [Angular CLI
README](https://github.com/angular/angular-cli/blob/master/README.md).
```

4.2 Configurazione applicativa

Il portale legge i dati dei contenuti informativi da delle API json i cui endpoint sono definiti in file config la cui struttura è quella che segue

```
{
    "globals": {
        "urlQuestionario": "http://www.comune.fi.it/export/sites/retecivica/comune_
firenze/sondaggi/questionario-suap.html",
        "denominazioneComune": "Comune di Gualdo Tadino",
        "denominazioneSportello": "Sportello Unico per le Attività Produttive",
        "denominazioneRegione": "Regione Umbria",
        "linkRegione": "http://www.regione.umbria.it/",


        "canaliSocial": [{
                "tipo": "twitter",
                "link": "#"
            },
            {
                "tipo": "facebook",
```

```
                "link": "#"
            },
            {
                "tipo": "youtube",
                "link": "#"
            }
        ],

        "footerLinks": [{
                "titolo": "Crediti",
                "link": "#"
            },
            {
                "titolo": "Note legali",
                "link": "#"
            },
            {
                "titolo": "Privacy policy e cookie",
                "link": "#"
            },
            {
                "titolo": "Contatti",
                "link": "#"
            }
        ]
    },

    "serviziRegionali": {
        "baseUrl": "https://<ip_api_json>/suap-accettatore/public_json/",
        "alias": "D612",
        "software": "SS"
    },

    "backend": {
        "baseUrl": "https://<ip_api_json>/areariservata2/public_json/",
        "alias": "DEF",
        "software": "SS",
        "urlVisura": "https://<ip_api_json>/stc-mobile-services/services/rest",

        "areaRiservata": {
            "baseUrl": "https://<ip_api_json>/areariservata/",
            "urlLogin": "Login.aspx?idComune={alias}&software={software}",
            "urlRegistrazione": "Registrazione.aspx?idComune={alias}&software={soft
ware}",
            "urlNuovaDomanda": "https://<ip_api_json>/suap-
accettatore/C113/nuovadomanda",
            "urlLeMiePratiche": "https://<ip_api_json>/suap-
```

```
accettatore/C113/lemiepratiche",
            "urlArchivioPratiche": "reserved/ArchivioPratiche.aspx?idComune={alias}
&software={software}",
            "urlScadenzario": "reserved/Scadenzario.aspx?idComune={alias}&software=
{software}",
            "urlDownloadModelloDomanda": "Public/ModelloDomanda/ModelloDomandaHandl
er.ashx?idComune={alias}&software={software}"
        }
    }

}
```