Creazione progetto da template, customizzazione e produzione war (versioni non standard)

Questa guida ricalca in parte questa (che è dedicata alle versioni rilasciate standard). Si differenzia in quanto permette di generare un war a partire da versioni del framework *speciali* o comunque *non standard*. Queste versioni NON sono rilasciate ufficialmente. Ciò comporta che non esse non sono ne taggate nel repository git, ne dispiegate su *Artifactory*. Un esempio di necessità di queste versioni speciali, può essere il caso in cui il rilascio sul cliente, contingentato esternamente nei tempi, prevede la presenza di nuove funzionalità, che magari verranno rilasciate solo in una futura release stabile del framework, in tempi ignoti successivi (previa fase di test, etc..). Non utilizzando materiale dispiegato su Artifactory, il build del war avviene tramite il build diretto del codice sorgente. Questo codice sorgente, che fa riferimento ad uno specifico branch del framework (e quindi, indirettamente ad una version), va quindi preventivamente scaricato in locale. Il nome del branche del sorgente sarà tipicamente qualcosa del genere *support/4.5-SPACE*, dove *support/* è fisso, *4.5* si riferisce alla version dove si presume che le feature contenute verranno rilasciate e considerate stabili, *-SPACE* è variabile.

Prima di procedere, ci si deve assicurare di aver installato nel proprio pc sia Apache Maven (Installazione di Maven) sia Git (Installazione di Git).

La guida sottostante può essere utilizzata in tutto od in parte a seconda si parta da zero o si voglia per esempio rigenerare il war in seguito al rilascio di nuove funzionalità sul branch support di commessa.

Gli esempi sottostanti partiranno dal presupposto che esistà un branch di commessa chiamato con il pattern: **[customer_code]+'_'+[module_name]** Per esempio: *cst_space*.

Prima di partire occorre conoscere il nome esatto che è presente nel branch di commessa in tutti i *pom.xml* interessati, dentro il tag <version>:

```
<version>4.5-SPACE</version>
```

a noi interessa

```
4.5-SPACE
```

Passaggio 1: Creazione struttura cartelle di base

Deve esistere la seguente struttura di cartelle:

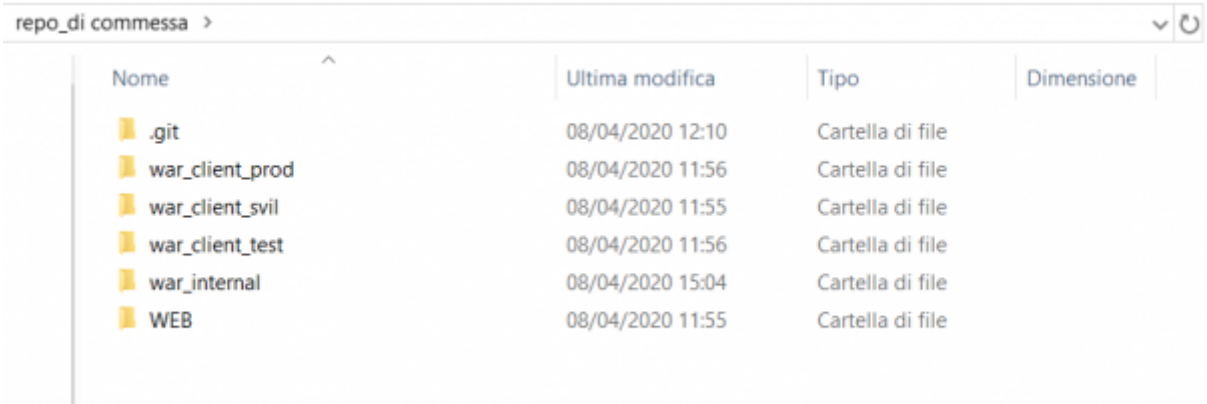
- *cst_space geowebframework project internal_war warname **buildpom&makegwwar.bat *file*** **makegproj.bat *file*** **pom.xml *file*** svil_war test_war prod_war **.git *file*** **.gitignore *file*** **README.md *file*** **La folder** cst_space**, può avere un nome qualsiasi, ma per convenzione verrà chiamata come il repository di commessa. La folder** geowebframework**,** NON VA CREATA MANUALMENTE**, ma verrà generata in automatico dal buildpom&makegwwar.bat, una volta opportunamente configurato. La folder** project **sarà il vero e proprio repository di commessa, il nome è fisso (in quanto usato a fuoco nel buildpom&makegwwar.bat). Si è scelto di tenere la folder del repository di commessa separata da geowebframework in via prudenziale. La presenza di un**

repository git dentro un altro repository git potrebbe condurre a problematiche in ambienti Linux, anche in caso di .gitignore configurato. I file *.git*, *.gitignore*, *README.md* sono tipici e denotano che questa è la folder del repository di commessa. Sotto *project* possono esserci un numero variabile di folder, in base alla necessità. In genere si prevede una folder per l'ambiente di test interno alle infrastrutture GeowebItalia, internal_war**, ed una o più folder per la produzione dei war nei vari ambienti, il cui numero è legato al cliente ed agli accordi preso con esso. Proseguiamo ora supponendo di voler generare il file .war dell'ambiente interno:** internal_war**. ===Passaggio 2: Generazione della struttura di base del war (esecuzione del file makeproj.bat) === La prima cosa da fare per creare un progetto di base nel proprio pc, è cliccare sul seguente link e avviare il download della cartella ZIP**

makegproj.zip

. Questa cartella contiene un file BAT, che una volta lanciato richiede due parametri: * il nome del nuovo progetto da generare, nel nostro esempio warname**, * la versione di GeowebFramework. Qui sceglieremo la stessa version che sarà disponibile nel codice sorgente presente nella folder** geowebframework**, quindi nel nostro esempio, *4.5-SPACE* (**e non una scelta tra quelle disponibili su Artifactory**) Dopo aver decompresso makegproj.zip e salvato il file makeproj.bat nel cartella che dovrà ospitare il nuovo progetto (nel nostro esempio *internal_war*), eseguirlo con un doppio click. In questo modo verrà aperta una finestra di *shell* che mostrerà l'avanzamento del download.**

Alla fine del processo, la finestra si chiuderà automaticamente, e si potrà vedere la cartella con il nome del progetto scelto (cioè quello inserito precedentemente in makeproj.bat). Quest'ultima conterrà tutti i file di configurazione e le librerie necessarie. Il file BAT, quindi, è utilizzato *una tantum*, cioè solo nella fase iniziale della creazione del nuovo progetto. ===Passaggio 3: Personalizzazione struttura e dei file di configurazione === ===Passaggio 4: Personalizzazione struttura e dei file di configurazione === A questo punto, il responsabile di commessa deve decidere la struttura specifica da dare al progetto: si devono individuare gli ambienti necessari, che corrispondono ai diversi WAR, e creare, per ognuno di essi, una sotto-cartella. Un esempio di contenuto standard di un repository di commessa è riportato in figura:



In linea generale, quindi, si può dire che la cartella di repository deve contenere: * un'unica cartella WEB per i contenuti statici dato che, solitamente, sono gli stessi per tutti gli ambienti; *

una cartella per l'ambiente interno; * una cartella per ogni ambiente esterno. Inoltre, le cartelle di ogni ambiente devono contenere tutti i file di configurazione necessari e devono rispettare la struttura standard. Generalmente, per ogni cartella di ambiente, va modificato il file pom.xml**, in cui deve essere aggiunta la giusta versione di GeowebFramework (la stessa utilizzata nel makegproj.bat) nel relativo tag: <code xml><com.geowebframework.version>4.4.3</com.geowebframework.version></code> Nel** pom.xml **vanno inserite anche le dipendenze a eventuali altri plugin di Geoweb necessari all'esecuzione del proprio progetto, come ad esempio: <code xml><dependency> <groupId>com.geowebframework</groupId> <artifactId>ispplugin</artifactId> <version>1.0.0</version> </dependency></code> In aggiunta, devono essere eventualmente modificati anche alcuni file di configurazione. In particolare, nel percorso *[nome_progetto]\src\main\resources*\ ci sono: * Il file** configuration.properties**, che raccoglie tutti i dati configurabili del progetto, che rappresentano entità costanti, come ad esempio i dati di accesso al database; anche in questo caso, ogni parametro è memorizzato come una coppia di stringhe, una contiene il nome del parametro (cioè la chiave) e l'altra memorizza il valore. * Il file** log4j.properties**, che contiene tutte le informazioni (codificate come coppie chiave-valore) necessarie alla scrittura dei log di Maven. Il log, infatti, è uno strumento molto utile che va configurato nel modo corretto: esso raccoglie gli output dei diversi processi e, analizzando questi output, permette di capire cosa è accaduto in caso di errori. * Il file** remapConfiguration.properties **che ha lo scopo di mappare i nomi delle proprietà all'interno del** configuration.properties **con nomi di variabili d'ambiente o di sistema. * Il file** webconfig.ini **che serve per alcune configurazioni della piattaforma Mapguide. Invece, nel percorso *[nome_progetto]\src\main\webapp\WEB-INF*\ si trovano: * Il** dispatcher-servlet.xml**, che serve da controller per le applicazioni web basate su Spring. Il contenuto è standard, ma, in particolari ambienti, alcuni tag non vengono usati e devono essere cambiati. * Il file** web.xml**, che descrive l'esecuzione dell'applicazione web e contiene configurazioni per l'accesso ai database. * Infine, il file** spring-security.xml**, che raccoglie informazioni di autenticazione e di connessione al metadata-source.

Le modifiche apportate ai vari file dipendono dal progetto, ma anche dall'ambiente in cui il progetto deve essere utilizzato.

Passaggio 3: produzione del WAR

Terminate tutte le operazioni di configurazione e personalizzazione del progetto, nella cartella di ogni ambiente di cui si vuole produrre il WAR, bisogna assicurarsi di aver copiato anche una coppia di file auto-eseguibili (cioè con estensione BAT e SH) chiamati makegwar e presenti tra quelli scaricati inizialmente da Artifactory. Se questi file non sono stati scaricati con il procedimento precedentemente descritto, è possibile fare il download della seguente cartella compressa, che li contiene entrambi:

makegwar.zip

.

Questi file si occupano di collezionare tutte le configurazioni e di produrre il WAR completo e dispiegabile. In realtà, per la creazione del WAR di un ambiente, va eseguito solo uno dei due file, ovvero makegwar.bat per pc con sistema operativo Windows, mentre makegwar.sh per pc con

sistema operativo Linux. Per eseguire il file, come prima, basta fare un doppio click: si aprirà una finestra di *shell* che mostrerà l'avanzamento dell'operazione. Al termine della procedura, nella finestra verrà riportato un messaggio di successo se tutto è andato a buon fine, oppure un messaggio di errore in caso di eventuali problemi. Il file WAR viene automaticamente salvato all'interno della sotto-cartella target nella cartella dell'ambiente scelto.