

## Rhino to WrmI STL istruzioni d'uso

Questo plugin è stato scritto in python versione 2 per essere compatibile con con le versioni di Rhinocers dalla 7 in avanti.

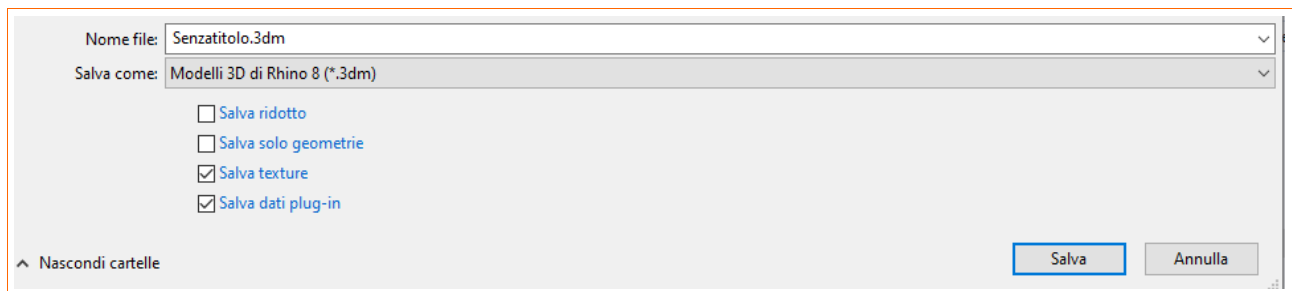
Si occupa dell'esportazione degli oggetti presenti sui livelli di rhino in maniera automatica in modo da nominare i file ottenuti in maniera coerente con il nome del file di partenza, del livello nel quale si trova l'oggetto da esportare, della versione del file.

Lo scopo è quello di velocizzare e rendere automatiche quelle operazioni ripetitive che tolgono tempo alla modellazione e possono facilmente generare errori.

Se si lancia il plugin da un nuovo file, che quindi non è mai stato salvato, verrà chiesto di salvarlo in una qualsiasi posizione locale

Un esempio di questo si ha quando importate gli oggetti da un altro sw, o quando li copiaincollate da una diversa sessione di rhino in un file vuoto.

Difficilmente partendo a modellare da zero in un nuovo file di rhino andrete tanto avanti nella modellazione da avere necessità di lanciare il plugin senza avere mai salvato il file di rhino.



La posizione di partenza sarà quella dell'ultimo file salvato da rhino.

I file Stl e WRML creati saranno salvati nella stessa posizione del file \*.3dm

È importante il nome che darai al file perchè gli oggetti presenti nel tuo filesaranno salvati tenendone conto.

Questo sarà determinante per mantenere ordine nelle varie versioni del tuo lavoro.

### Tasto "Esc"

Alcuni processi possono causare loop molto lunghi o infiniti, in questo caso per interrompere l'esecuzione dello script premi il tasto **ESC** della tua tastiera.

Probabilmente hai per errore esploso la polisuperficie o la mesh generando un numero molto alto di superfici o poligoni.

Se questo capitasse per altri motivi, cerca di riprodurre l'errore e di capirne il motivo, mandami una mail , magari allegando il tuo file, così posso metterlo a posto.

## Creazione del nome dei file esportati:

I nomi dei file esportati saranno formati da

[parte del nome del file 3dm prima dell'ultimo "\_"] [numero] ["\_" + nome del livello]

## Regole principali per la nominazione del file di rhino:

Tu hai piena libertà di nominare il tuo file, ma per essere utilizzato al meglio questo script richiede che tu segua alcune semplici regole.

Caratteri speciali come "< > : " / \ | ? \* "ecc . verranno ignorati nel nome dei file esportati, si presume che siano errori di battitura! In molti casi sarà lo sistema ad impedirvi l'inserimento di determinati caratteri perchè entrerebbero in conflitto con il sistema operativo stesso.

Tenendo conto di questo possiamo idealmente dividere il nome del file in tre parti

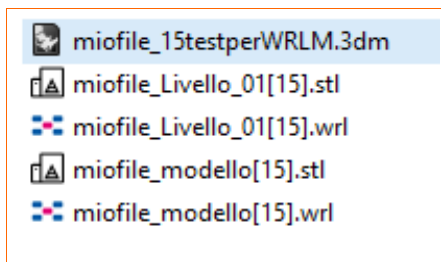
[qualsiasi testo compresi "\_" al suo interno][\_][numero][qualsiasi testo tranne "\_"]

ad esempio

nominando il file :

**miofile\_15testperWRML.3dm**

verranno salvati i seguenti file:



Notate che tutto quello che segue "\_15" viene ignorato nel nome dei file creati. Questo è utile per creare una copia del file rhino da utilizzare per l'esportazione dei wrml senza "rovinare" il file di rhino.

**miofile\_15testperWRML.3dm** potrebbe essere una copia modificata per l'esportazione del file **miofile\_15.3dm** che quindi rimane intatto.

Pensate ad esempio ad un oggetto che necessita dell'applicazione di una texture su una parte della sua superficie, che è quindi diviso e non è più una polisuperficie chiusa.

Tornerò su questo più avanti.






Il numero dopo l'ultimo "\_" viene utilizzato per il numero della versione

Da notare che l'underscore utilizzato per separare la parte da ignorare dal nome vero e proprio è l'ultimo a destra... per questo la parte dopo il numero non deve contenere nessun simbolo di underscore.

Ad esempio un nome come questo :

**miofile\_15test\_per\_WRML.3dm**

i file esportati saranno:

|  |                  |
|--|------------------|
|  miofile_15test_per_Livello_01[0].stl | 05/11/2025 17:31 |
|  miofile_15test_per_Livello_01[0].wrl | 05/11/2025 17:31 |
|  miofile_15test_per_modello[0].stl    | 05/11/2025 17:31 |
|  miofile_15test_per_modello[0].wrl    | 05/11/2025 17:31 |
|  miofile_15test_per_WRLM.3dm          | 05/11/2025 17:31 |

il **WRML** presente nel nome del file **.3dm** dopo l'ultimo \_ è stato ignorato mentre non essendoci alcun numero dal quale ricavare la versione è stata aggiunto " [0]".

## Creazione del nome dei file esportati:

I nomi dei file esportati saranno formati da

**parte del nome del file .3dm prima dell'ultimo "\_" + \_nome del livello + [numero dopo "\_"]**










negli esempi precedenti gli oggetti sono presenti nei livelli "livello01" e "modello"

Il formato wrml viene utilizzato per la stampa 3d a colori in quanto mantiene i colori e le textures applicati all'oggetto .

Il colore dell'oggetto può essere applicato al livello o direttamente all'oggetto.

Il colore applicato all'oggetto prevale rispetto a quello applicato al livello.

Possono esserci più oggetti presenti sullo stesso livello nel qual caso i nomi degli oggetti esportati dal livello avranno un numero progressivo.

|  |                  |
|--|------------------|
|  miofile_Livello_01_2[15].stl | 05/11/2025 17:48 |
|  miofile_Livello_01_2[15].wrl | 05/11/2025 17:48 |
|  miofile_Livello_01[15].stl   | 05/11/2025 17:48 |
|  miofile_Livello_01[15].wrl   | 05/11/2025 17:48 |
|  miofile_Livello_01_1[15].stl | 05/11/2025 17:48 |
|  miofile_Livello_01_1[15].wrl | 05/11/2025 17:48 |
|  miofile_modello[15].stl      | 05/11/2025 17:48 |
|  miofile_modello[15].wrl      | 05/11/2025 17:48 |
|  miofile_15testperWRLM.3dm    | 05/11/2025 17:48 |

Nell'esempio il livello "Livello\_01" presenta 3 oggetti.

## Oggetti processati e oggetti ignorati nell'esportazione:

Il plugin esporterà nei formati wrml e stl solo gli oggetti visibili al momento dell'esportazione, sia che siano selezionabili che bloccati.

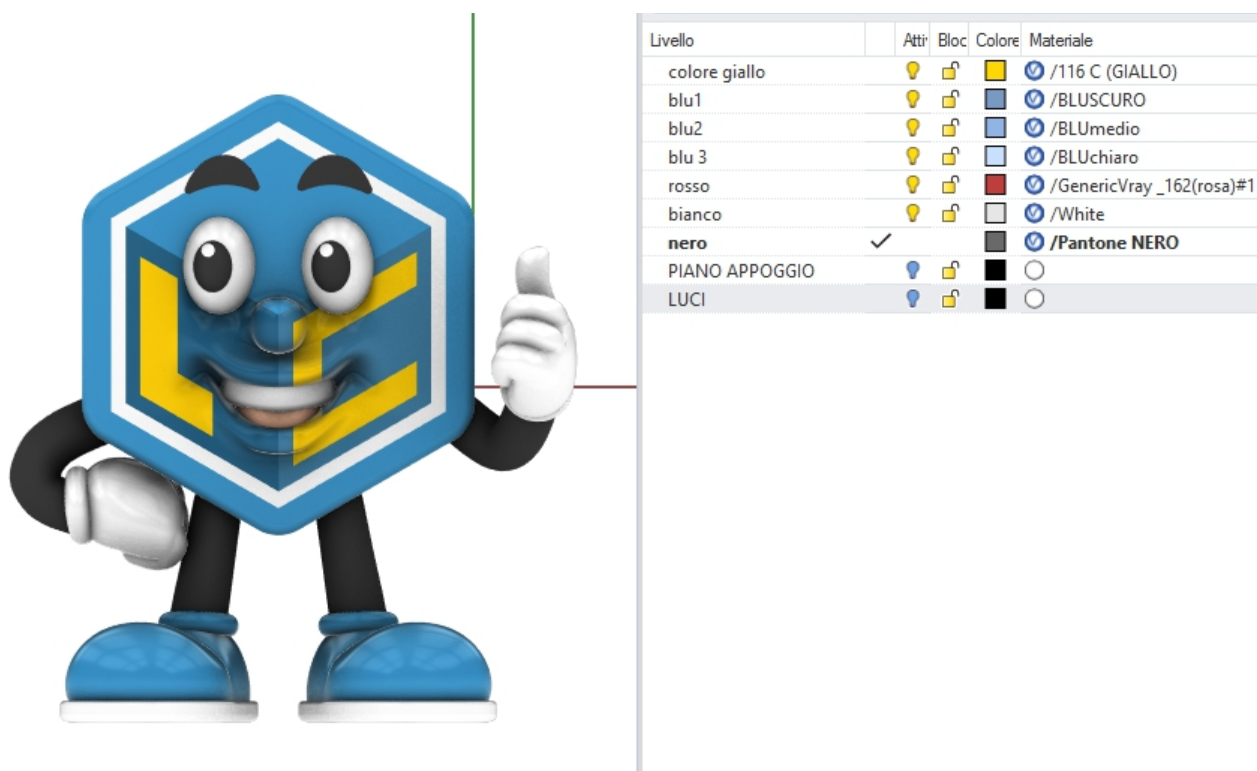
Gli oggetti bloccati ma visibili saranno preventivamente sbloccati.

I livelli invisibili saranno ignorati con tutto quello che si trova al loro interno.

## Organizzazione dei file sui livelli per una corretta esportazione

Per l'utilizzo ottimale del plugin conviene organizzare il file dividendo gli oggetti su livelli e

applicare il colore principale al livello.



In questo esempio il modello del personaggio è destinato al costampaggio, ogni parte del modello che rappresenta il colore presenta al suo interno i canali per il passaggio del colore.

È organizzato in modo da avere ogni parte sul livello nominato secondo il colore in modo che poi le parti riportino questo nel proprio nome.

Lanciando il plugin su questo file saranno generate 7 coppie di file , i livelli LUCI e PIANO APPOGGIO sono spenti e quindi verranno ignorati

|                                     |                  |
|-------------------------------------|------------------|
| Personaggio3d_nero[15].wrl          | 05/11/2025 18:34 |
| Personaggio3d_nero[15].stl          | 05/11/2025 18:34 |
| Personaggio3d_bianco[15].wrl        | 05/11/2025 18:34 |
| Personaggio3d_bianco[15].stl        | 05/11/2025 18:34 |
| Personaggio3d_rosso[15].wrl         | 05/11/2025 18:34 |
| Personaggio3d_rosso[15].stl         | 05/11/2025 18:34 |
| Personaggio3d_blu_3[15].wrl         | 05/11/2025 18:34 |
| Personaggio3d_blu_3[15].stl         | 05/11/2025 18:34 |
| Personaggio3d_blu2[15].wrl          | 05/11/2025 18:34 |
| Personaggio3d_blu2[15].stl          | 05/11/2025 18:34 |
| Personaggio3d_blu1[15].wrl          | 05/11/2025 18:34 |
| Personaggio3d_blu1[15].stl          | 05/11/2025 18:34 |
| Personaggio3d_colore_giallo[15].wrl | 05/11/2025 18:34 |
| Personaggio3d_colore_giallo[15].stl | 05/11/2025 18:34 |
| Personaggio3d_15.3dm                | 05/11/2025 18:14 |

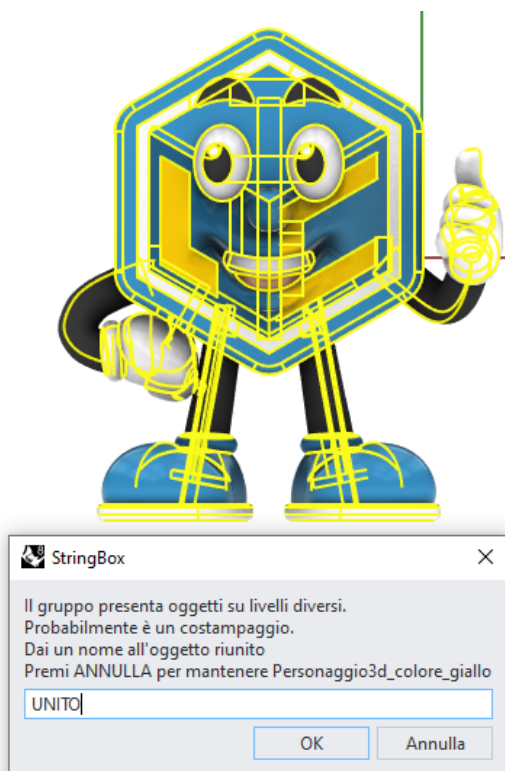
Sebbene questa esportazione sia utile per creare i vari passaggi di costampaggio, potrebbe essere utile avere un unico file wrml per la stampa di un prototipo.

Per fare questo basta unire tutte le parti che formano un oggetto "finale" unico in un gruppo.

## Presenza di oggetti raggruppati:

In presenza di gruppi il plugin si comporta in maniera diversa se gli oggetti raggruppati si trovano tutti sullo stesso livello o su livelli diversi.

Se gli oggetti sono tutti sullo stesso livello, il gruppo viene considerato come un oggetto quindi prenderà il nome del livello con un eventuale numero progressivo in presenza di altri gruppi o oggetti sul medesimo livello.



Se almeno un oggetto raggruppatto si trova su un livello diverso verrà invece chiesto di salvarlo con un diverso nome, questo perché è presumibile che si tratta di un costampaggio

Il nome proposto in alternativa è quello del livello che si trova più in alto nell'albero dei livelli

I file **Personaggio3d\_UNITO[15].wrl** e **Personaggio3d\_UNITO[15].stl** verranno aggiunti ai file

In presenza di gruppi con oggetti su livelli diversi e contemporaneamente di oggetti liberi... questi avranno un numero progressivo, mentre per il livello raggruppatto verrà proposto di rinominarlo o di chiamarlo con il nome livello seguito da un numero progressivo più alto per non entrare in conflitto con i precedenti.

Nel caso su un livello ci siano più gruppi bisogna anche in questo caso distinguere se gli oggetti presenti

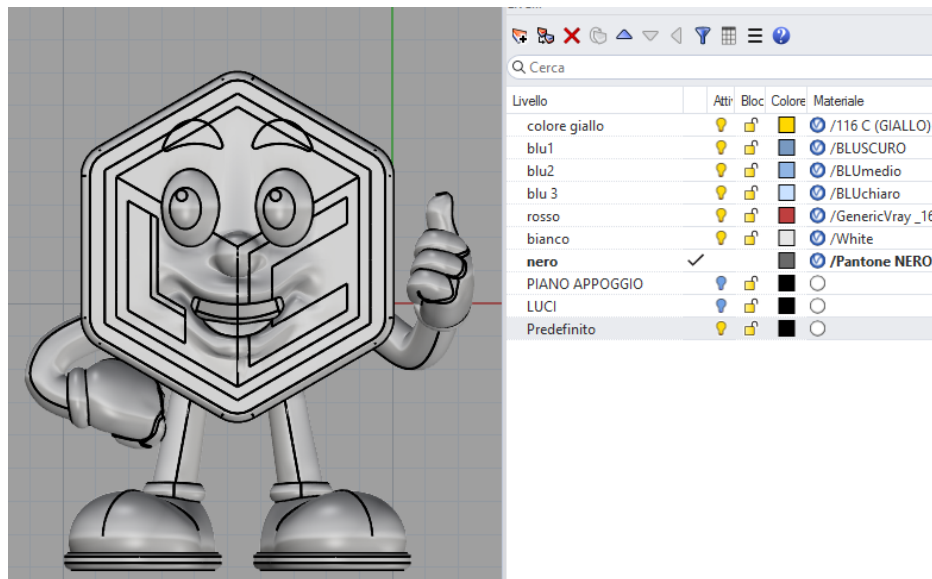
nel gruppo sono tutti sullo stesso livello, nel qual caso prenderà il nome del livello, eventualmente seguito da un numero progressivo, o se gli oggetti raggruppati sono multilivello, nel qual caso verrà chiesto di rinominare il file.

In ogni caso bisogna fare attenzione nel rinominare il file in quanto se il nome generato risultasse uguale ad un altro, verrà sovrascritto (verrà fatta una copia di backup, ma solo del precedente! Quindi occhio!)

Raggruppare gli oggetti che si trovano su livelli diversi, se il colore è applicato sul livello, può essere molto utile in una fase del lavoro precedente a quella finale, dove non sono ancora presenti i canali per il passaggio dei colori negli stampi ma si vuole comunque ottenere il file di un oggetto unico in formato wrml per la stampa a colori.

In questo esempio il modello è formato da una polisuperficie chiusa sul livello "predefinito"

Ho già creato l'albero dei livelli con il colore per le varie parti, colore applicato al livello.



*il personaggio LIBBY in una unica polisuperficie chiusa*

Estraendo le superfici relative ad un colore e spostandole sul relativo livello, raggruppandolo in un unico gruppo e lanciando il plugin, verrà creata una copia di file in formato wrml e stl del personaggio unito e colorato.

Essendo un gruppo con parti su livelli diversi, verrà chiesto di rinominare il file in alternativa a quello proposto.

Si noti che non è necessario avere una unica polisuperficie chiusa per ottenere una coppia di file validi per la stampa e quindi utilizzabili in questa versione preliminare.

Le varie parti (leggi colori) possono essere oggetti che si compenetrano tra di loro.. se questi vengono disposti sui livelli relativi al colore, o hanno un materiale applicato, e vengono raggruppati, il plugin comunque esporterà una coppia di file.

In questo caso però la possibilità che il file esportato presenti degli errori è maggiore.










È buona pratica passare i file esportati in un software di esportazione che nella maggior parte dei casi provvederà ad eliminare gli errori.

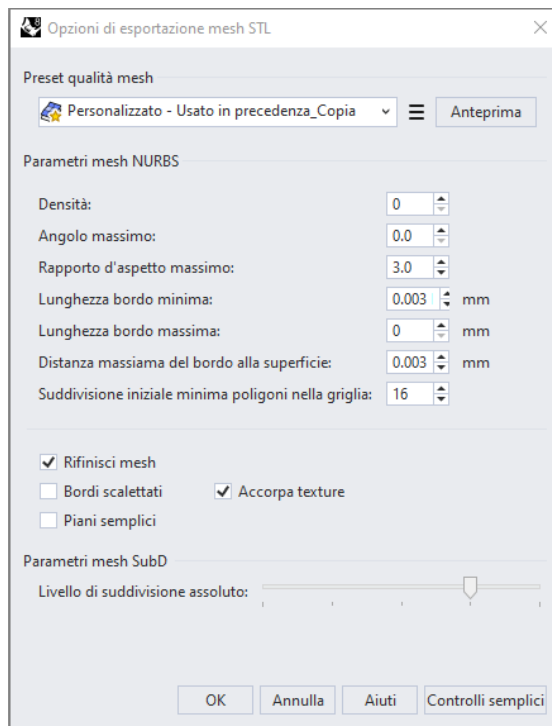
## Parametri per l'esportazione Stl e WRML

Il plugin permette l'esportazione di qualsiasi geometria presente sui livelli di Rhino nei formati wrml e stl che sono formati poligonali.

Gli oggetti creati o importati in formato poligonale, vengono esportati tali e quali senza riduzione o ottimizzazione del numero dei poligoni.

Gli oggetti con geometria Nurbs, superfici, polisuperfici e SubD vengono trasformati in MESH poligonali in fase di esportazione secondo questi parametri:

|  |                  |                 |           |
|--|------------------|-----------------|-----------|
|  SFERE_SFERA_SUBD_da_mesh[2].wrl        | 06/11/2025 10:06 | File WRL        | 31.754 KB |
|  SFERE_SFERA_SUBD_da_mesh[2].stl        | 06/11/2025 10:06 | Oggetto 3D      | 12.401 KB |
|  SFERE_Sfera_SUBD_da_superfici_4[2].wrl | 06/11/2025 10:06 | File WRL        | 2.796 KB  |
|  SFERE_Sfera_SUBD_da_superfici_4[2].stl | 06/11/2025 10:06 | Oggetto 3D      | 1.101 KB  |
|  SFERE_sfera_Mesh[2].wrl                | 06/11/2025 10:06 | File WRL        | 116 KB    |
|  SFERE_sfera_Mesh[2].stl                | 06/11/2025 10:06 | Oggetto 3D      | 47 KB     |
|  SFERE_sfera_nurbs[2].wrl               | 06/11/2025 10:06 | File WRL        | 116 KB    |
|  SFERE_sfera_nurbs[2].stl               | 06/11/2025 10:06 | Oggetto 3D      | 47 KB     |
|  SFERE_2.3dm                            | 06/11/2025 10:04 | Rhino 3-D Model | 7.759 KB  |



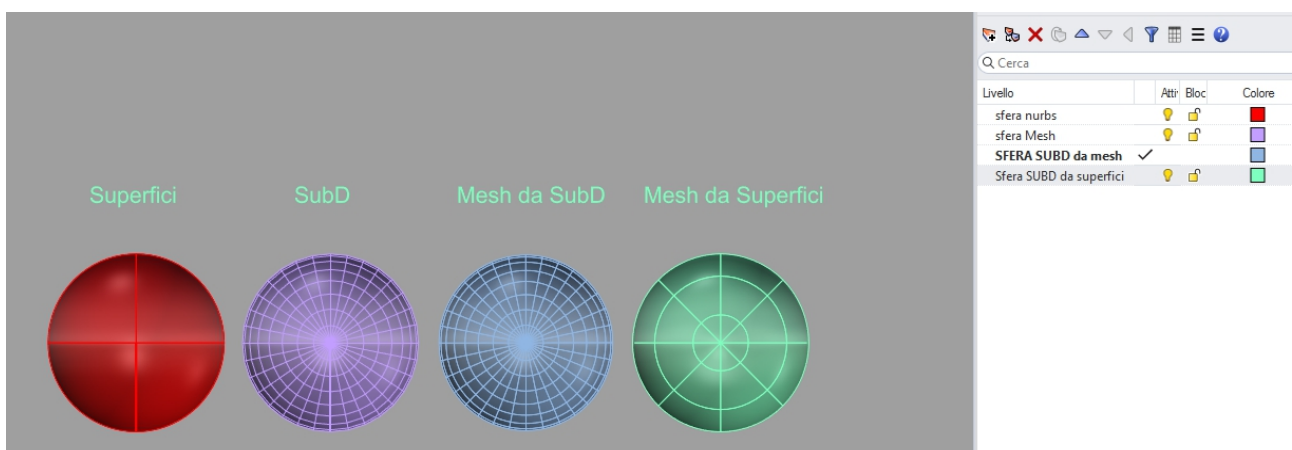
Questi valori non sono modificabili e garantiscono un'ottima approssimazione della superficie mantenendo relativamente basso il peso del singolo file .

Un caso particolare si può avere nel caso di esportazione di geometrie SUBD se questi vengono creati direttamente trasformando MESH.

Comando: ASubD

**Selezione mesh, superfici ed estrusioni:**

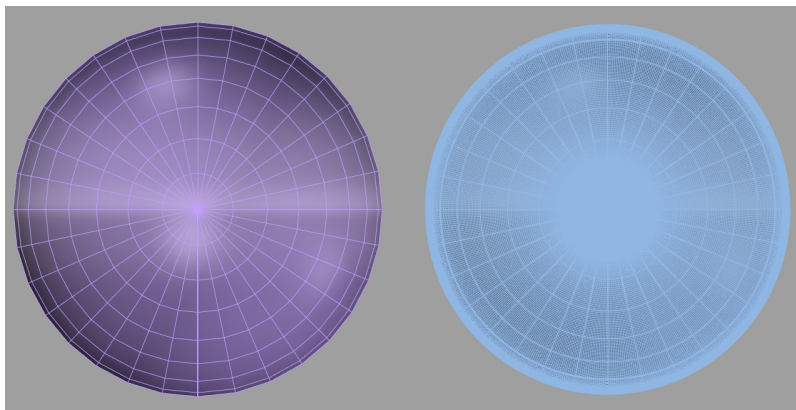
In quanto la mesh creata in fase di esportazione sarebbe estremamente fitta e quindi estremamente pesante.



A titolo di esempio : una semplice sfera creata come superfici, come mesh, trasformando la mesh in SubD e trasformando la superficie in SubD genererà le seguenti coppie di file:



Notate il peso in KB della sfera creata da superfici che pesa 116Kb, identica a quella mesh ( il processo di esportazione di fatto trasforma la superficie in mesh) e quella creata dalla trasformazione della mesh in SubD che pesa 31754 Kb.



Se zoomate per visualizzare la mesh creata vedrete il motivo dell'aumento del peso: la fittissima rete di poligoni creata per approssimare la superficie.(a sx la mesh e a dx quella generata dalla geometria Mesh>SubD).

### **Presenza di più geometrie sullo stesso livello.**

Come già spiegato il plugin rileva la presenza di più oggetti presenti sullo stesso livello ed assegna un numero progressivo per evitare conflitti e quindi la sovrascrittura del file.

Nell'esempio delle sfere in presenza di tre oggetti sullo stesso livello avremo

**SFERE\_sfera\_nurbs\_2[2].stl e SFERE\_sfera\_nurbs\_2[2].wrl**

**SFERE\_sfera\_nurbs\_1[2].stl e SFERE\_sfera\_nurbs\_1[2].wrl**

**SFERE\_sfera\_nurbs[2].stl e SFERE\_sfera\_nurbs[2].wrl**

il numero più alto indica l'oggetto più vecchio.

Questo processo avviene anche nel caso le geometrie siano nascoste e quindi non esportate.

Nell'esempio precedente, se visibile solo la sfera più vecchia verrà esportato solo

**SFERE\_sfera\_nurbs\_2[2].stl e SFERE\_sfera\_nurbs\_2[2].wrl**

### **Buone regole da seguire :**

1:creare un albero dei livelli coerente con gli oggetti da esportare.

2:assegnare il materiale base al livello, eventuali altri materiali o texture possono essere assegnati direttamente all'oggetto

3:mettere gli oggetti sui livelli giusti in modo che ne ereditino il materiale.

4:tenere visibili solo gli oggetti da esportare.

Eventuali geometrie di prova , note,pianidiritaglio , piani infiniti geometrie di supporto che non devono essere esportate...devono essere nascosti. Per quanto spiegato precedentemente riguardo al numero progressivo assegnato in presenza di più oggetti sullo stesso livello, sarebbe opportuno



creare un livello apposito dove spostare tutta la geometria "di supporto" che verrà spento e quindi ignorato prima dell'esportazione.

4:controllare la geometria da esportare: una polisuperficie esplosa genererà tante coppie di file quante sono le superfici... questo potrebbe richiedere tempo e incasinarvi la directory di lavoro.

5: se questo fosse indispensabile , ad esempio se su alcune superfici deve essere assegnata una mappatura e un materiale diverso dal resto dell'oggetto, occorre raggruppare l'oggetto completo. Questo verrà letto dal plugin come singolo oggetto (vedi **Presenza di oggetti raggruppati**)

## Tasto ESC: uscita forzata dal plugin

Può capitare di lanciare il plugin per sbaglio, non avere notato la presenza di geometrie inutili avere esploso per errore una polisuperficie o una mesh ecc. Per evitare di riempire la directory di lavoro con migliaia di file e aspettare pazientemente la fine del processo... è possibile interrompere forzatamente l'esecuzione del plugin tenendo premuto il tasto Esc.

## Conclusioni:

Questo plugin ottimizza l'esportazione della geometria da rhino nei formati Stl e WRML che sono i formati standard per la stampa 3d e per la realizzazione degli stampi per l'industrializzazione dei prototipi.

Il salvataggio dei file con un nome coerente con il nome del file , che tenga traccia della versione del modello(se il nome del file è scritto tenendo conto di alcune semplici regole) con la mesh generata ottimizzata potrebbe velocizzare molto il vostro lavoro automatizzando le operazioni ripetitive.

Potreste non averne bisogno se non avete la necessità di creare molte versioni del lavoro da presentare al committente, se avete un solo oggetto su un solo livello, se non usate rhino in maniera professionale.

In tutti gli altri casi noterete un notevole risparmio di tempo e un ordine nei file generati, tempo che potete dedicare alla modellazione e ordine che vi permetterà di condividere i vostri file con meno possibilità di errori.

Scaricando il file avrete la possibilità di testare il plugin senza alcuna limitazione per 30 giorni dopodiché sarà necessario acquistare la licenza per un anno.

## File importati

Il plugin è stato scritto in python 2.# in modo da essere compatibile con rhino dalla versione 7 in avanti.

Funziona con Rhino, non è standalone, ed è in grado di esportare tutti gli oggetti compatibili con rhino.

Io normalmente modello per superfici direttamente in rhino ma potrebbe essere comodo utilizzarlo anche solo in fase di esportazione.

Ad esempio file che richiedono una texturizzazione 3d , come la pelliccia di un animale, potrebbero essere difficili da creare direttamente in Rhino; Potresti fare la base in superfici in rhino e applicare la texture con un programma di terze parti (come ad esempio 3dcoat), importare il modello nuovamente in rhino, magari su un livello apposito e usare il plugin per la distribuzione finale.

O potresti essere abituato a lavorare direttamente in poligoni con un altro software , come zbrush, ma dovere comunque passare i file in rhino per conformità con le necessità del committente. In questo caso importando i file poligonali che compongono le parti del modello useresti il plugin per l'esportazione finale dopo avere creato l'albero dei livelli con il materiali e i nomi.

## Installazione

Scarica la cartella e salvala dove vuoi sul tuo disco, trascina il file **.yak** dentro una sessione aperta di rhino...fatto!

Dovrebbe comparire la barra dei comandi con il pulsante per avviare il plugin

Spostala dove vuoi e agganciala alle barre degli strumenti per ritrovarla al successivo riavvio di Rhino.

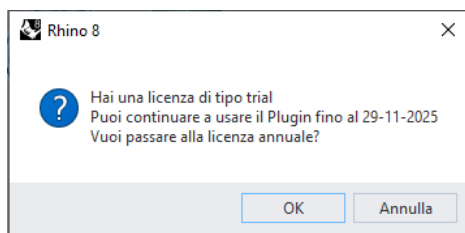


In alternativa puoi installarlo da Opzioni/plugin/installa in questo caso ti verrà richiesto il file RHP

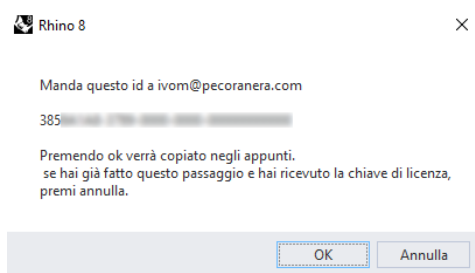
**Al primo avvio** del plugin ti verrà chiesto se vuoi testare il plugin o acquistare la licenza annuale.

Questo messaggio comparirà all'avvio del plugin, ma solo una volta al giorno per non essere troppo invasivo, con il countdown dei giorni rimanenti per la versione di prova

In caso voleste passare alla versione a pagamento bisogna dare l'ok nel box di dialogo



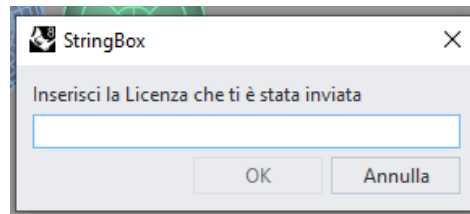
si aprirà un nuovo box:



premendo ok, l'id del tuo pc/mac verrà copiato negli appunti

manda una mail a : [ivom@pecoranera.com](mailto:ivom@pecoranera.com) con la prova del tuo acquisto.incolla l'ID dagli appunti, metti il tuo nome . Mettendo nel campo Oggetto "richiesta di licenza annuale" mi faciliti l'organizzazione delle mail in arrivo.

Verificato l'acquisto ti rimanderò indietro la chiave seriale personalizzata che dovrai inserire nel box accessibile premendo "annulla"nel box precedente.



Da questo momento potrai usare il plugin per un anno.

Se sei arrivato fino qui per errore, premendo annulla continuerai ad usare la licenza di prova per il tempo rimanente.

### Comandi da riga di comando.

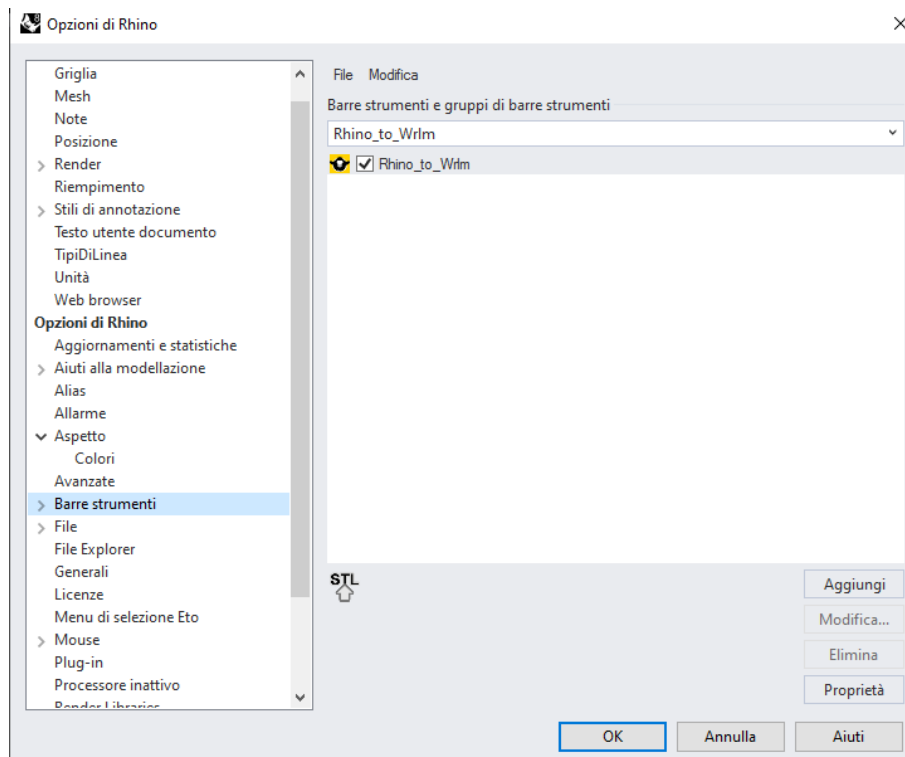
Come ogni operazione eseguibile in Rhino, è possibile fare partire il plugin direttamente da riga di comando, questo è utile nel caso le barre di comando del plugin sia sparita, cosa non impossibile se usate le barre libere e non fissate.

Scrivendo "**\_RhinoToWrmlStl**" o "**RhinoToWrmlStl**" (a dir il vero bastamettere le prime lettere, c'è il completamento automatico) e dare invio per fare partire il plugin

È anche un modo sicuro per vedere se il plugin è in esecuzione o è stato disinstallato.

Se per qualche motivo vi è sparito il pulsantino per fare partire il plugin e siete sicuri che questo è installato, per farlo ricomparire aprite:

opzioni/aspetto/barre applicazioni/RhinoToWrml



Mettete la linea di spunta e il pulsante ricomparirà.

### Disinstallazione:

digita "gestorepacchetti" nella riga di comando, nella finestra "pacchetti installati" seleziona RhinoToWrml e clicca su disinstalla.

