Come creare nuovi kit con weidu (versione 0.9, di al17)

Programmi necessari:

- Word
- Weidu
- Near Infinity
- Dltcep (opzionale)

Premessa: nelle prossime pagine troverai tanti zero e tanti uno. Meglio mettersi gli occhiali se hai problemi di vista! :) Passando alla vera premessa: questo è il secondo tutorial che scrivo su come creare nuovi kit. Quello vecchio è sempre valido, e può sostituire il paragrafo due di questo.

1) Cosa non si può fare

La creazione di nuovi kit per Baldur's Gate ha diversi limiti. Di seguito possiamo leggere quelli che mi vengono in mente.

Non possiamo porre limitazioni all'utilizzo di determinate armi, armature e oggetti vari. Tuttavia possiamo sfruttare le restrizione dei kit già esistenti (come il berserker o il kensai). Così facendo però avremo *tutte* quante le sue restrizioni. Possiamo riempire un slot del personaggio con un oggetto finto, con la caratteristica “undroppable” (non rimuovibile), per impedire al nostro personaggio di utilizzare un determinato tipo di oggetti (amuleto, anelli, cintura, elmo, armatura, etc).

Non possiamo creare nuovi kit per maghi (a dire il vero possiamo, ma dobbiamo sfruttare gli slot per il ladro, e comunque è un lavoro molto complicato che necessita di un tutorial tutto suo), monaci, barbari e maghi selvaggi.

Non possiamo creare un numero infinto di kit per una classe, ma solo "quelli che si vedono nella schermata di creazione del personaggio”. Il limite è nove. Tuttavia questo limite può essere superato rendendo determinati kit accessibili a non tutte le razze (quindi, sette razze per nove slot, uguale sessantatre).

Non possiamo mettere pulsanti nella barra inferiore estranei alla classe originale, quindi niente scacciare non morti per i ladri e scassinare per i sacerdoti. Possiamo però simulare quasi tutti gli effetti attraverso le "innate abilities" (pulsante stella, vedi file clabpr03.2da).

Se non sei un guerriero non puoi memorizzare incantesimi da mago, se sei un ladro non puoi memorizzare quelli da sacerdote, e cosi via (tuttavia puoi aumentare il numero di incantesimi memorizzabili per livello o sbloccarli innanzi tempo).

2) I file .2da (o la teoria)

La maggior parte dei parametri di un kit (razze consentite, minima costituzione, massima costituzione, biclassabile in, etc) sono racchiusi nei file .2da, modificabili con un editor di testo (word, notepad, context, etc). Analizziamoli uno a uno...

kitlist.2da

```
--------------
     ROWNAME        LOWER MIXED HELP  ABILITIES PROFICIENCY  UNUSABLE   CLASS
...
20   HELM           25196 25169 25222 CLABPR03  48           0x02000000 3
--------------
```

20: un numero di cui importa davvero poco.

HELM: nome di riferimento del kit (se non lo avessimo ancora capito, prendiamo come esempio il sacerdote di Helm,)

25196: string del nome *tutto in minuscolo* del kit (sacerdote di helm). Anche di questo ci importa poco.

25169: string del nome con iniziali in maiuscolo del kit (Sacerdote di Helm). Anche di questo ci importa poco.

25222: string della descrizione del kit (SACERDOTE DI HELM: I seguaci del dio neutrale degli Osservatori e dei Protettori sono guerrieri a tutti gli effetti e spesso appaiono come difensori degli innocenti...). Anche di questo ci importa poco.

CLABPR03: nome del file .2da (clabpr03.2da) dove sono indicate tutte le magie, sia quelle innate (pulsante stella nel gioco), sia quelle che garantiscono bonus o malus particolari al kit che possiamo dare solo tramite file .spl (resistenza al fuoco, ai danni da taglio, +X alla CA o al thac0, cinture finte, etc). Questi file dei kit originali sono tutti preceduti da un "clab", più le iniziali della classe (“pr” sta per “priest”, sacerdote), più una numerazione a due cifre diversa per ogni kit. Io sconsiglio vivamente di seguire questo sistema, onde evitare possibili ripetizioni con kit aggiunti da altri mod. Possiamo scegliere il nome che vogliamo per il file, quindi inventiamone uno particolare, ad esempio: pippo001 (non dobbiamo mai superare gli otto caratteri). Cosi chiamandolo, il kit farà riferimento al file pippo001 e non a quello clabpr03.

48: se scopri a cosa funziona, fammelo sapere! L'unica cosa certa è che non ci devono essere due kit con lo stesso numero. Comunque, come per gli altri numeri, a noi non interessa.

0x02000000: limitazioni agli oggetti. Qui dobbiamo copiare il parametro di un kit già esistente (berserker: 0x00000001, kensai: 0x00000004... sono troppi, consulta il file originale). Nota: se selezioniamo 0x00000004 (kensai) il nostro kit sarà anche in grado di brandire una eventuale arma riservata ai soli kensai.

3: fa riferimento alla classe originale del kit (guerriero: 2, paladino: 6, ranger: 12, ladro: 4, bardo: 5, druido: 11, sacerdote: 3, mago: 1).

abclasrq.2da

	MIN_STR	MIN_DEX	MIN_CON	MIN_INT	MIN_WIS	MIN_CHR
...						
HELM	0	0	0	0	9	0

HELM: nome di riferimento del kit (questa sarà una costante, quindi è l'ultima volta che mi ripeto).

I numeri sono in ordine: minima forza, destrezza, costituzione, intelligenza, saggezza e carisma.

In molti dei kit originali troviamo "zero" come caratteristica minima. Nei nostri kit invece dobbiamo inserire come minimo un bel "3" (così siamo sicuri di evitare possibili problemi).

Nota: avere troppi limiti minimi alti è un vantaggio per il kit. Saremo infatti sicuri che il nostro personaggio avrà almeno X in determinate caratteristiche e per le altre un paio di lanci di dadi dovrebbero risolvere il problema.

abclsmod.2da

```
                        MOD_STR MOD_DEX MOD_CON MOD_INT MOD_WIS MOD_CHR
...
HELM                    0       0       0       0       0       0
--------------
```

I numeri sono in ordine: modificatore alla forza, destrezza, costituzione, intelligenza, saggezza e carisma.

Per aumentare di X, basta sostituire lo zero con la X nella colonna corrispondente alla caratteristica desiderata. Se vogliamo diminuire di X, basta sostituire con -X.

Al massimo possiamo abbassare una caratteristica di -8, e aumentarla per non superare 25 come totale. Per diminuire più di otto o per creare un personaggio con una caratteristica fissa dovremo usare un file .spl (guarda clabpr03.2da).

abdcdsrq.2da

```
--------------
                        MIN_STR MIN_DEX MIN_CON MIN_INT MIN_WIS MIN_CHR
...
HELM                    0       0       0       0       17      0
--------------
```

Inserisci pure tutti 1, se vuoi (comunque devi aggiornare anche questo file). Il abdcdsrq.2da non ha alcun effetto sul nostro kit. Teoricamente dovrebbe indicare le caratteristiche minime per biclassare *a* questo kit, ma come ben sappiamo, non possiamo biclassare a un kit.

Allora a cosa serve questo file? Se noti, in alto, ci sono anche i valori per le classi pure.

abdcscrq.2da

```
--------------
                        MIN_STR MIN_DEX MIN_CON MIN_INT MIN_WIS MIN_CHR
...
HELM                    0       0       0       0       15      0
--------------
```

Questo file indica il punteggio minimo della caratteristica primaria (o delle caratteristiche primarie) per biclassare *dal* nostro kit. Di solito si imposta 15.

Le caratteristiche principali sono: guerriero -> forza, mago e simili -> intelligenza, sacerdoti -> saggezza, ladro -> destrezza.

alignmnt.2da

```
--------------
            L_G     L_N     L_E     N_G     N_N     N_E     C_G     C_N     C_E
...
HELM        0       1       0       0       1       0       0       1       0
--------------
```

In questo caso i numeri non indicano più una quantità. Lo zero sta semplicemente per falso, e l'uno sta per vero.

Qui dobbiamo scegliere gli allineamenti consentiti al nostro kit. Le iniziali della prima riga stanno per:

L_G: legale buono
L_N: legale neutrale
L_E: legale malvagio
N_G: neutrale buono
N_N: neutrale puro

N_E: neutrale malvagio
C_G: caotico buono
C_N: caotico neutrale
C_E: caotico malvagio

Nota: devi sempre segnare almeno un allineamento come 1 (vero).

Guardando il .2da relativo al nostro sacerdote di Helm, possiamo verificare quello che già sappiamo: solo personaggi neutrali (legali, neutrali puri o caotici).

clabpr03.2da (o pippo001.2da)

	1	2	3	4	5	...
ABILITY1	GA_SPCL731	****	****	****	****	
ABILITY2	GA_SPCL732	****	****	****	****	
ABILITY3	****	****	****	****	****	
...						

Eccoci arrivati al file di riferimento delle magie. Le colonne indicano a quale livello la determinata magia verrà attivata, le righe a... beh, servono solo per aggiungere altre magie per stello livello!

Davanti ad ogni file dobbiamo inserire un "GA_" se la nostra magia è una abilità innata, lanciabile durante il gioco cliccando sul “pulsante stella” in basso a destra, o "AP_" se la nostra magia ci da un bonus un malus che non può essere impostato con i .2da.

GA_: il modo migliore per trovare una magia innata è fare un ricerca con near infinity. Ecco un esempio: andiamo in "search - spell" e battiamo “Imposizione delle Mani”. Compariranno diversi file. Come facciamo a sapere qual è quello giusto? Dobbiamo guardare quali "effect" ha, se lo "spell type" è "innate" e se l’"Ability icon location" di ogni "spell ability" è "Innate slots". Se uno solo dei tre non è impostato in questo modo, dobbiamo andare a verificare il file successivo. Nota: non tutte le magie hanno una versione che funziona come abilità innata. In questo caso dobbiamo farne una copia (della versione per maghi o sacerdoti), darle un nuovo nome (pippoma1.spl, ad esempio) e modificare gli ultimi due parametri come riportato ("spell type" e "Ability icon location").

AP_: per andare sul sicuro (io ho sempre fato così) impostiamo come sopra lo "spell type" e l'"Ability icon location". Fatto questo, tutti gli "effect" della nostra magia saranno dati come bonus (o malus) al kit. Una spiegazione dettagliata su come creare nuove magie per limitare i kit necessità di un altro tutorial. Se sei poco pratico di magie, scrivi a: rtt-mod@libero.it.

Infine, andiamo a controllare le due magie innate del kit del sacerdote di Helm, per verificare se sono le stesse che ben conosciamo. spcl731.spl: spada inquisitrice, spcl732.spl: vera vista... sono proprio loro! ;)

Nota: in SoA inizi almeno al livello 6-7, quindi dare un GA_ o un AP_ alla colonna del livello 1, 2 o 3 non cambia nulla.

clasweap.2da

	SMALL_SWORD	LARGE_SWORD	BLUNT	MISSILE	BOW	SPIKED
...						
HELM	0	0	1	1	0	1

Seleziona i tipi di armi consentite. Ovviamente 1 è vero (utilizzabile) 0 è falso (non utilizzabile).

weapprof.2da

```
--------------
                            ID      NAME_REF        DESC_REF    MAGE
LARGE_SWORD                 0       8668            9589        0
SMALL_SWORD                 1       8732            9590        1
...
--------------
```

A differenza degli altri file .2da il nostro kit non si trova scritto su una riga, ma sulla colonna più a destra.

Per ogni riga abbiamo un tipo di arma, e il numero sotto “HELM” indica la specializzazione massima raggiungibile con l’arma della medesima riga (da zero a cinque).

Gli “ID” (identificazione del gioco del tipo di arma) il “NAME_REF” (nome dell'arma) e il “DESC_REF” (descrizione del tipo di arma) non ci interessano.

Nota: se dai cinque stelline alla SMALL_SWORD (spada corta) a un kit per sacerdoti, il nostro personaggio non sarà comunque in grado di brandire il tipo di arma per le limitazione imposte dalla classe di origine.

*K_*_**.2da*

```
--------------
        KIT
1       0
2       1
3       2
4       3
5       31
--------------
```

In Baldur’s Gate 2 abbiamo diversi K_*_**.2da. Noi dobbiamo aggiornare tutti quelli che desideriamo come combinazioni possibili classe (*) – razza (**) per il nostro kit.

Come “*” abbiamo:

B: bardo
C: chierico
D: druido
F: guerriero
P: paladino
R: ranger
T: ladro

Come “**” abbiamo:

H: umano
D: nano
G: gnomo
E: elfo
HE: mezz’elfo
HL: halfling
HO: mezz’orco

Quindi, se creiamo un kit per guerrieri accessibili ai soli elfi e mezz’elfi, andremo a modificare i file K_F_E.2da e K_F_HE.2da.

Cosa dobbiamo fare? Dobbiamo aggiungere nella prima colonna il numero successivo all’ultimo (nell’esempio 5) e nella seconda il numero alla sinistra del nome di riferimento del nostro kit che troviamo in kitlist.2da (nell’esempio 31).

Alla fine, l’unica cosa importante che dobbiamo ricordarci sono le combinazioni K_*_** (weidu penserà al resto ;)).

25stweap.2da

```
--------------
                          ID      NAME_REF        DESC_REF     MAGE      FIGHTER
ARMOR                     0       8668            9589         *         CHAN09
SHIELD                    1       8732            9590         *         *
...
--------------
```

Questo è un .2da per ToB. In esso dobbiamo scrivere gli item che avrà un nuovo personaggio (cioè non importato da SoA).

Ignoriamo "ID", "NAME_REF" e "DESC_REF". Ad ogni righe, nella colonna del nostro kit (che come in weapprof.2da si trova a destra) dovremo scrivere il file di uno scudo in corrispondenza di “shield”, di un anello sotto “ring1”, etc. Se non vogliamo nessun oggetto, inseriamo un "*".

Proprio come per i “clab”, il modo più facile per trovare il file di un item è la funzione di ricerca di near infinity. Ovviamente possiamo creare anche nuovi oggetti e darli al nostro kit.

luabbr.2da

```
--------------
                        ABBREV
MAGE                     Ma0
FIGHTER                  Fi0
CLERIC                   Cl0
THIEF                    Th0
BARD                     Ba0
PALADIN                  Pa0
DRUID                    Dr0
RANGER                   Ra0
...
--------------
```

Nella prima colonna dobbiamo scrivere il nome di riferimento del kit, nella successiva, a seconda della classe originaria:

Ma0: mago
Fi0: guerriero
Cl0: chierico
Th0: ladro
Ba0: bardo
Pa0: paladino
Dr0: druido
Ra0: ranger

In questo modo diciamo al kit di fare riferimento al tabella del file LU***.2da (con ***: Ma0, Fi0, etc) per le abilità di alto livello (quelle di ToB).

Nota: è anche possibile creare nuova tabelle di abilità di alte livello. Il tutto però è un filo complicato e verrà spiegato (forse) in una futura versione del tutorial.

3) L'installazione con weidu (o la pratica)

Ora dobbiamo solo creare il .tp2 per il nostro mod.

Nell’esempio del .tp2 che si trova qualche riga più giù aggiorneremo tutti i file .2da di cui abbiamo parlato nel paragrafo due.

Sentiti libero di usarlo come base per il tuo kit mod (le frasi di spiegazione precedute da “//” possono non essere eliminate).

Esempio di file .tp2 (kitrtt.tp2 del mod RTT Kit pack)

```
--------------
// cartella necessaria per permettere la disinstallazione
BACKUP ~rttkit\backup~

// credits
AUTHOR ~Kit: Al 17 (rtt-mod@libero.it), weidu setup: al17 and the bigg~

// ogni .tp2 deve avere almeno un "BEGIN". Dopo aver lanciato il “setup-xxxx.exe” per l’installazione comparirà la frase tra “~”. A questo punto dovremo dare l’ordine di continuare o di chiudere
BEGIN ~Vuoi installare i nuovi kit?~

// nome di riferimento per i file .2da. Può anche essere completamente diverso dal nome vero del kit. Non superare gli undici caratteri per sicurezza
ADD_KIT ~palaelf~

// inizio aggiornamento righe e colonne dei file .2da. Il primo numero andrà ad aggiornare la prima colonna o la prima riga a seconda dei casi, il secondo numero andrà ad aggiornare la seconda colonna o riga, e così via. Consulta il paragrafo 2 del tutorial se hai dubbi sul .2da

// aggiunge linea a clasweap.2da
~palaelf                1          1          1          1          1
1          1          1    ~

// aggiunge colonna a weapprof.2da
~palaelf 2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2
2  2  2  2  2  2  2  2  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
~

// aggiunge linea a abclasrg.2da
~palaelf                3       3       6       3       16      17     ~

// aggiunge linea a abclsmod.2da
~palaelf                0       1      -1       0       0       0      ~

// aggiunge linea a abdcdsrq.2da
~palaelf                0       0       0       0       0       0      ~

// aggiunge linea a abdcscrq.2da
~palaelf                0       0       0       0       0       0      ~

// aggiunge linea a alignmnt.2da
~palaelf                1       0       0       0       0       0       0
0       0  ~

// aggiunge linea a dualclas.2da
~palaelf                0       0       0       0       0       0     ~

// file per le abilità innate e i bonus/malus speciali (i CLAB originali)
~rttkit/RTTKPA05.2DA~

// aggiunge il kit ai file K_*_**.2da
~ K_P_H ~
```

```
// restrizioni oggetti e valore per la classe originale (quelle del file
kitlist.2da)
~0x00000001 6~

// indica il file LU***.2da a cui deve far riferimento il kit (abilità di alto
livello di ToB)
~Pa0~

// aggiunge colonna a 25stweap.2da
~ CHAN09  *  HELM07  BAG26  RING06  RING31  *  BOOT01  AMUL20  BRAC10  BELT06
AROW11,40  BULL03,40  BOLT06,40  POTN52,5  POTN04,2  POTN14,5  HAMM07  SW1H27
STAF08 ~

// in ordine: scrivi il nome del kit tutto in minuscolo, scrivi il nome del kit
con le iniziali in maiuscolo, scrivi la descrizione del kit (“lower”, “mixed” e
“help” del file kitlist.2da)
SAY ~elfo paladino~
SAY ~Elfo Paladino~
SAY ~ELFO PALADINO: bla bla bla...~
--------------
```

Se il file .2da non fanno riferimento a .spl, .itm, .eff, etc non originali (vedi file clabpr03.2da e file 25stweap.2da), possiamo saltare direttamente al paragrafo quattro. Ma se così non è, dobbiamo aggiungere anche loro al nostro .tp2.

Farlo è molto facile. Ecco un esempio come prima:

```
--------------
// copia il file spad01.itm del nostro mod, posizionato in rttkit/item/, nella
cartella override
COPY ~rttkit/item/spad01.itm~ ~override/spad01.itm~

// Nome oggetto non identificato
SAY NAME ~Spada Lunga~

// Nome oggetto identificato
SAY NAME2 ~Spada Lunga +2: Il Torturatore~

// Descrizione oggetto non identificato
SAY UNIDENTIFIED_DESC ~Le spade lunghe sono comunemente usate...~

// Descrizione oggetto identificato
SAY DESC ~La Spada del Torturatore è stata creata dal mago elfico...~
--------------
```

La stessa cosa vale per i .spl. Invece per i .bam (animazioni/icone) non occorre aggiungere nessun "SAY". Alcune magie con “effect” particolari e i file .eff necessitano di ulteriori "SAY". In questi casi la cosa migliore da fare è utilizzare l'ordine di weidu "automate". Consulta il readme di weidu per le procedure esatte.

4) Consigli generali

Creare un kit è facile, creare un buon kit è molto difficile. Personalmente sconsiglio di creare kit strani... molto spesso infatti si cade nel ridicolo (cacciatore di criceti :)) o in kit extra-potenti (Mezzo Vampiro).

In generale, a ogni vantaggio dovrebbe corrispondere uno svantaggio per equilibrare il kit.

Meglio dieci kit ben fatti e personalizzati che quaranta anonimi.