Superglús - Introducción

Manual : Uto (Basándose en la trascripción de Siew del manual de PAW Spectrum)

Tutorial

Basado en el software PAW:

Programa: T. J. Gilberts, Graeme Yeandle y P. Wade

Manuales: T. J. Gilberts and Graeme Yeandle y Andrés Samudio

1. Introducción

Bienvenido al mundo del escritor de aventuras…

Superglús te ofrece todas las facilidades para producir aventuras con gráficos de alta calidad con la capacidad de ser ejecutadas en múltiples plataformas (Win32, Linux, Solaris, Amiga, etc.)

Superglús te proporciona una base estructurada para poder escribir el juego, pero sigue siendo tarea del escritor el tener un guión imaginativo e ideas originales.

Los manuales que vienen con Superglús cubren todos los aspectos de su uso. Este manual de introducción servirá como guía para crear una aventura y recomendamos su lectura detenida y el ejercicio de sus ejemplos antes de intentar hacer un juego propio. La guía técnica da una información detallada de todo el sistema y debe ser usado como libro de referencia al escribir una aventura.

2. Escribiendo una aventura

Enfoque previo

Es importante un buen planteamiento si se quiere crear una aventura con un nivel aceptable. De nada sirve sentarse a la máquina y empezar a teclear esperando la inspiración. Lo único que se conseguirá es un desorden de números y palabras y sin otro recurso que volver a empezarlo todo de nuevo.

Para ilustrar el método recomendado, crearemos y desarrollaremos una simple aventura desde el inicio de la idea hasta el resultado final.
¡Recuerda salvar tu juego con regularidad!

El guión inicial

Esa idea inicial es siempre la parte más difícil al crear cualquier cosa. Una historia original puede dar al juego un interés mucho mayor que la vieja historia de rescatar a la consabida princesa.

Guiones hay de todas clases y de todo tipo, pero aconsejamos que si intentas basar la historia en una película, libro o comic, y luego intentas usarla comercialmente, te asegures antes de obtener el permiso del autor o de los actuales dueños del copyright.

En nuestro ejemplo tomaremos los problemas que pueden ocurrirle a un pasajero de vuelta a su casa:
Mientras estaba en la parada del autobús, el billete de subida fue arrebatado por un golpe de viento y luego cogido y llevado por un pajarito hasta un cercano parque.

El ordenador juega la parte del pasajero a quien se debe dirigir para encontrar el BILLETE antes de que llegue el autobús.

Diseño del juego

Ahora que la idea está más clara es importante hacer un boceto del área de juego, en nuestro ejemplo sería algo parecido al **diagrama número 2**.

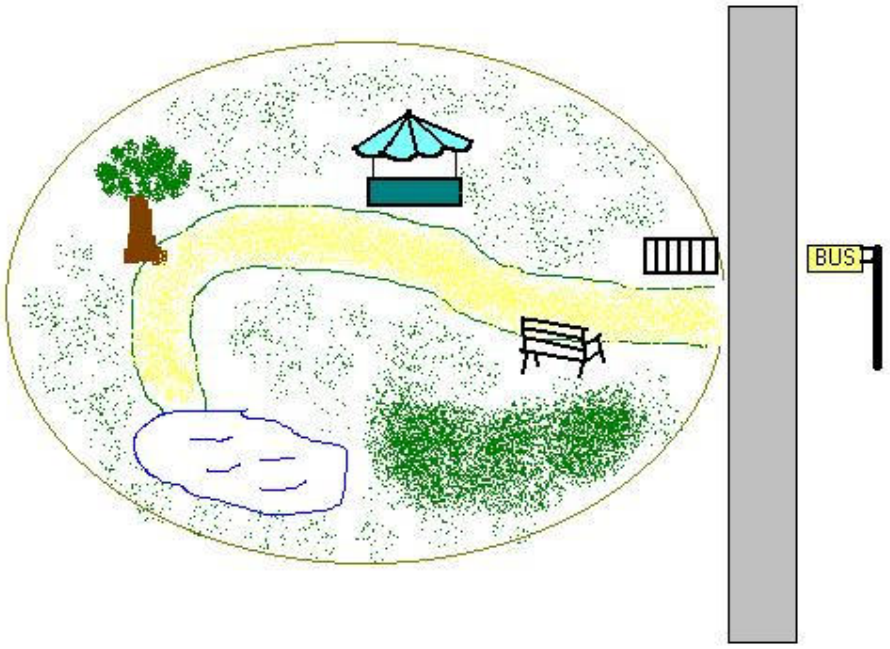


diagrama 2

Es de notar que la zona de juego debe estar cerrada de una manera lógica, o el jugador no entenderá por qué no puede ir en una dirección si no hay nada que impida el paso.

Para el ordenador, una aventura consiste en un determinado número de apartados o localidades que el jugador debe visitar, es ahora el momento de decidir qué zonas del boceto se tomarán como localidades y numerarlas individualmente.

Se debe intentar hacer una escala consistente o lógica (a menos que el juego sea ilógico intencionadamente), evitando que con un solo paso se vaya de una localidad a otra que aparentemente queda a varias millas de distancia.

La localidad 0 debe ser reservada siempre como pantalla de títulos y dejaremos libre también la 1 para darle un uso especial, por lo tanto comenzaremos desde la 2 hacia arriba.

En nuestro ejemplo hemos escogido 7 localidades:

- 2- parada del autobús
- 3- en el jardín
- 4- cerca del banco
- 5- en el pabellón de música
- 6. el estanque
- 7- al lado del árbol
- 8- arriba del árbol

Ahora podemos empezar las descripciones de las localidades. Hay que procurar mantener una misma forma de verbo todo el tiempo, generalmente primera(yo) o segunda(tu) persona, o pronto el jugador tendrá una seria crisis de identidad. Sea cual sea la forma elegida, debe de estar de acuerdo con los Mensajes del Sistema (ver mas adelante).

Para claridad y ver mejor los posibles movimientos haremos el mapa del **diagrama 3**

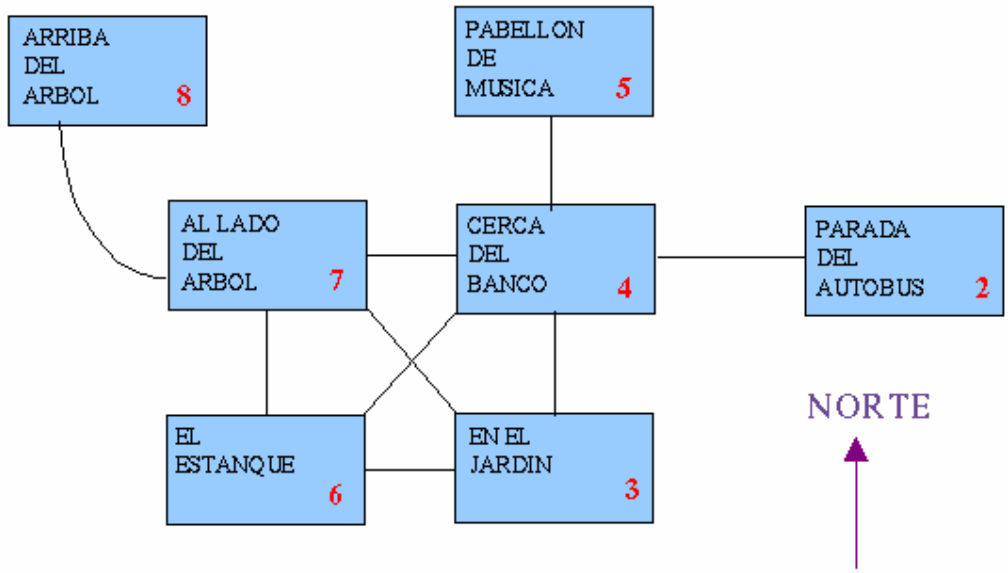
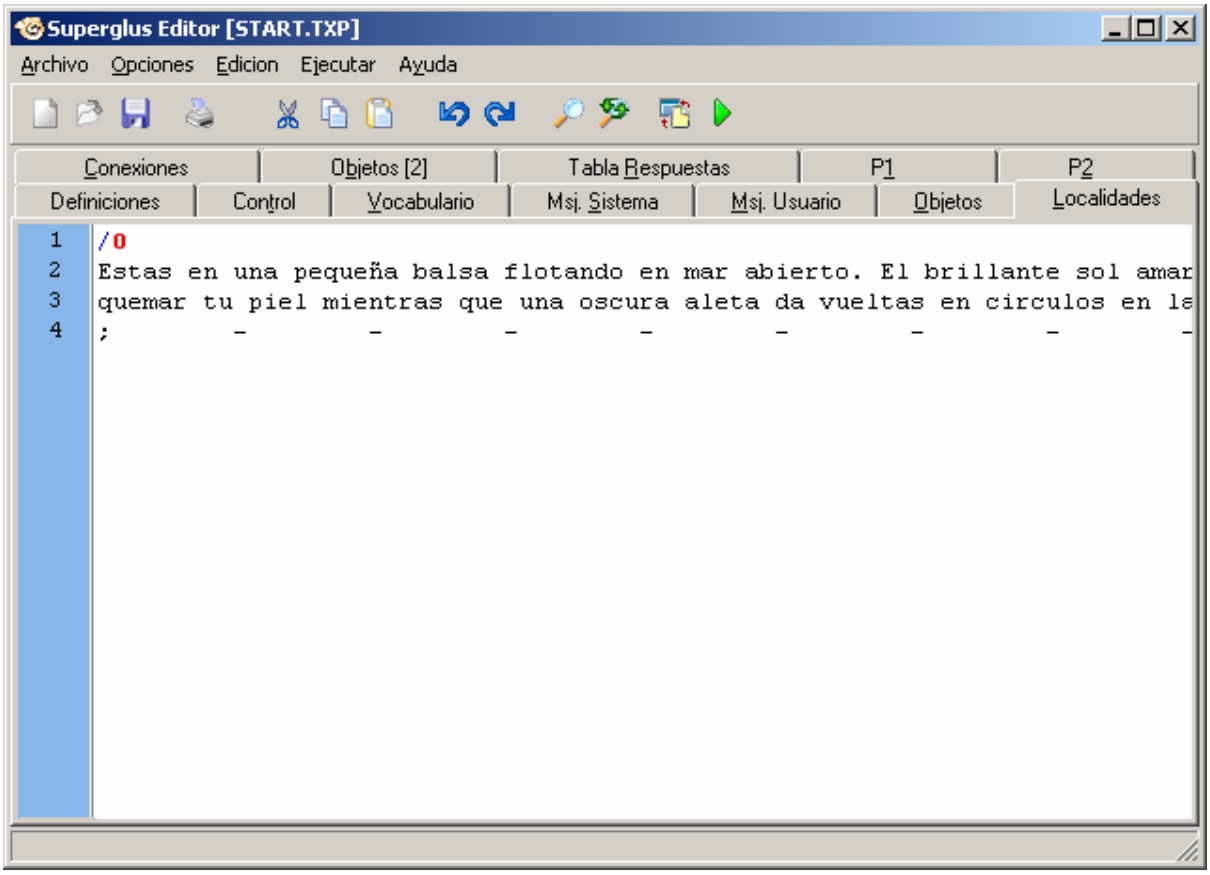


diagrama 3

3. Empezando a teclear

Localidades

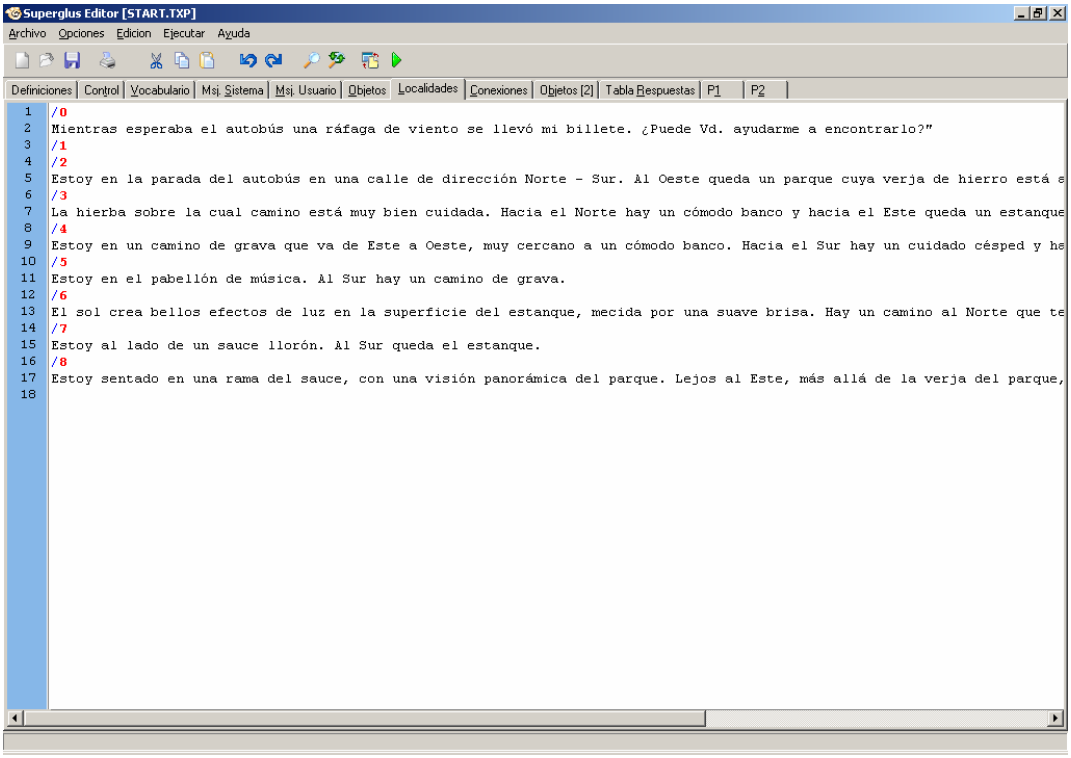
Selecciona la opción Alt-L en el menú principal del editor de Superglús, o bien accede a la sección LTX con tu editor favorito si no lo usas.



Substituye el texto "Estas en una pequeña balsa flotando en mar abierto. El brillante sol amarillo comienza a quemar tu piel mientras que una oscura aleta da vueltas en circulos en la…" por "Mientras esperaba el autobús una ráfaga de viento se llevó mi billete. ¿Puede Vd. ayudarme a encontrarlo?"

Luego añade

/1
/2
Estoy en la parada del autobús en una calle de dirección Norte - Sur. Al Oeste queda un parque cuya verja de hierro está abierta
/3
La hierba sobre la cual camino está muy bien cuidada. Hacia el Norte hay un cómodo banco y hacia el Este queda un estanque.
/4
Estoy en un camino de grava que va de Este a Oeste, muy cercano a un cómodo banco. Hacia el Sur hay un cuidado césped y hacia el Norte hay un pabellón de música.
/5
Estoy en el pabellón de música. Al Sur hay un camino de grava.
/6
El sol crea bellos efectos de luz en la superficie del estanque, mecida por una suave brisa. Hay un camino al Norte que termina en un lloroso sauce. Al este queda el cuidado césped.
/7
Estoy al lado de un sauce llorón. Al Sur queda el estanque.
/8
Estoy sentado en una rama del sauce, con una visión panorámica del parque. Lejos al Este, más allá de la verja del parque, puedo ver la parada del autobús.



Como podéis ver hemos dejado la localidad 1 sin ningún texto.

Conexiones

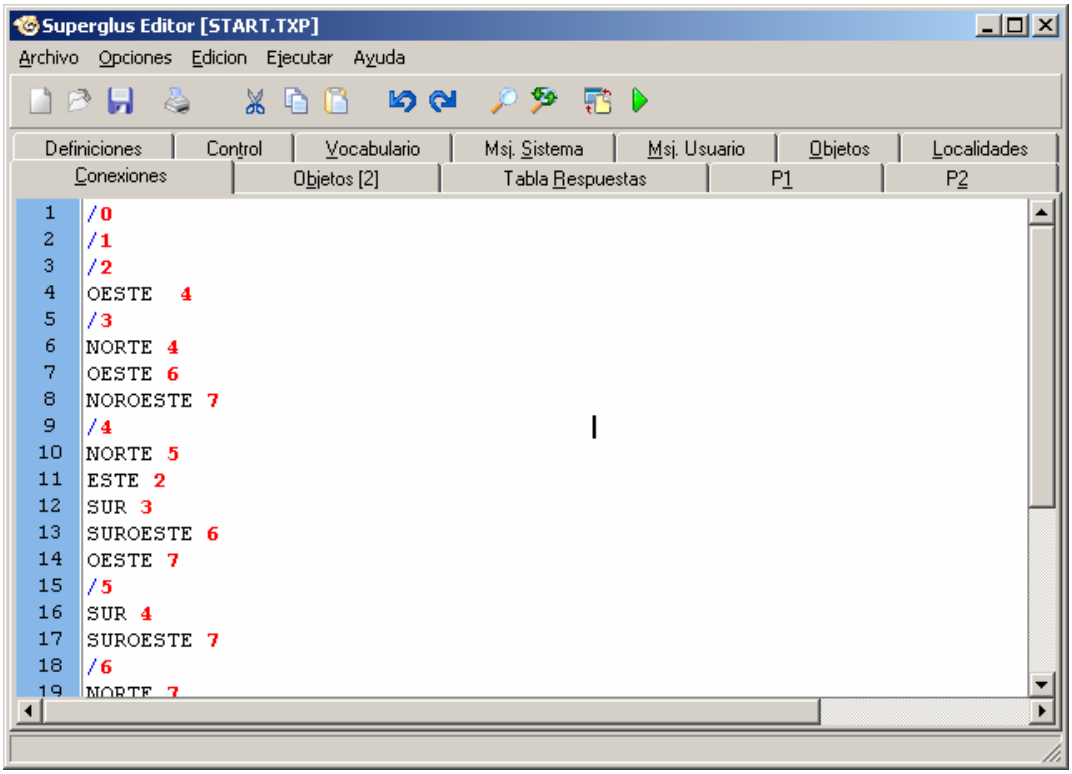
En el apartado CON o en la solapa conexiones en el Editor de Superglús accedemos a un listado similar al de las localidades, pero para dar de alta las conexiones.

Incluiremos y modificaremos el listado para que sea como esto:

/0
/1
/2
OESTE 4
/3
NORTE 4
OESTE 6
NO 7
/4
NORTE 5
ESTE 2
SUR 3
SO 6
OESTE 7
/5
SUR 4
SE 7
/6
NORTE 7
NE 4
ESTE 3
/7
ARRIBA 8
NE 5

ESTE 4
SE 3
SUR 6
/8
BAJAR 7

Como veis se trata de una simple lista de conexiones de cada localidad seguidas de la localidad a la que lleva dicha conexión.



De momento vamos a poner una conexión falsa desde la localidad 0 para que se pueda acceder a la localidad 2, la localidad inicial. Substituimos el solitario /0 por:

/0

NORTE 2

4. Jugando el juego

Ha llegado el momento de intentar probar nuestro juego. Para ello pulsamos F5 en el editor o compilamos y ejecutamos desde la línea de comandos (como hacer esto está descrito en la guía técnica).

Nota: Es importante haber configurado el editor para que todo funcione correctamente, en el manual del Editor tenéis información sobre como configurarle para empezar a funcionar.

Ahora debe de aparecer la pantalla con el título que tecleamos al comienzo. La línea de INPUT se usa para poner las órdenes para que Superglús las interprete y ejecute tus acciones de acuerdo con la información que le hayas dado cuando escribiste el juego.

De momento, solamente le hemos dicho hacia donde nos puede llevar cuando una dirección se teclea. Eso es lo que intentaremos probar. Ahora empecemos el juego correctamente tecleando NORTE, desde la localidad 0, y por supuesto, ENTER.

Veremos entonces que la pantalla se limpia y que la descripción para la localidad 2 aparece. Si ello no ocurre, probablemente la entrada en las conexiones está equivocada. No te preocupes, porque se puede volver otra vez al editor tecleando ABANDONAR o RETIRAR y respondiendo S a la pregunta que el se te hace y contestando N a la siguiente pregunta de que "si quieres probar otra vez" el juego o no.

Vuelve a las conexiones y corrige lo necesario.

Ahora hay que probar a moverse entre las localidades viendo todos los movimientos posibles (toma nota de cualquier movimiento erróneo para poderlo corregir cuando vuelvas al Editor). También es tiempo de hablar de algunas órdenes que Superglús ya tiene dentro y que conoce. Por ejemplo, tecleando M o MIRAR se repite otra vez la descripción de la localidad, lo que es muy útil para el jugador si hay un montón de texto en una descripción que ya ha hecho SCROLL.

Tecleando I o INVENTARIO harás una lista de los objetos que llevas en este momento; por ahora llevarás solamente un objeto desde el comienzo, pero no podrás hacer nada con el.

5. Objetos

Objeto es cualquier cosa que el jugador puede manipular dentro del juego, por ejemplo: una manzana que puede comer, una llave que se puede usar para abrir una puerta, una espada para pelear, etc.

En nuestro juego, que es bastante simple, usaremos los siguientes objetos (no todos ellos tendrán una función en el juego finalizado):

Objeto 0 - una antorcha
Objeto 1 - una bolsa
Objeto 2 - un emparedado
Objeto 3 - una manzana
Objeto 4 - un billete de autobús
Objeto 5 - una piedra
Objeto 6 - un anorak

Podemos definir estas descripciones e objetos en el apartado OTX o en la solapa ‘Objetos’ en el Editor:

```
/0
una antorcha
/1
una bolsa
/2
un emparedado
/3
una manzana
/4
un billete de autobús
/5
```

una piedra
/6
un anorak

Pasamos a la solapa ‘Objetos 2’ del editor o al apartado OBJ si usamos un editor externo e incluimos los siguiente:

```
/0  254    1    ANTORCHA     _     10000000000000000000000000000000 00000000000000000000000000000000
/1  2      3    BOLSA        _     00000000000000000000000000000000 00000000000000000000000000000000
/2  254    1    BOCADILLO    _     00000000000000000000000000000000 00000000000000000000000000000000
/3  254    1    MANZANA      _     00000000000000000000000000000000 00000000000000000000000000000000
/4  8      1    BILLETE      _     00000000000000000000000000000000 00000000000000000000000000000000
/5  3      1    PIEDRA       _     00000000000000000000000000000000 00000000000000000000000000000000
/6  253    3    ANORAK       _     01000000000000000000000000000000 00000000000000000000000000000000
```

Aquí se definen una serie de cosas importantes para los objetos: cada línea comienza por el número de objeto, seguida de la localidad inicial, que debe ser un numero de localidad existente o bien una de las tres localidades especiales:

Esta opción tiene varios valores especiales que son muy importantes (son localidades que no existen):

- 252 es para objetos no creados, por ejemplo, que no existen todavía dentro del juego o no son visibles hasta que no se haga algo.
- 253 tiene todos los objetos llevados encima (puestos) por el jugador.
- 254 tiene todos los objetos llevados por el jugador, pero no puestos encima.

Le sigue el peso relativo: un peso relativo es un peso que viene referenciado por una medida que tu decides. Pueden ser kilos, pueden ser libras o pueden ser lo que tu quieras, incluso medidas de peso inventadas. El caso es que hay una manera después de decirle a Superglús cuantas unidades de dicho peso puede llevar el jugador, así que tu decides según el jugador que sea. En este caso hemos considerado una medida inventada, siendo el jugador capaz de llevar 8 unidades de dicha medida (valor por defecto).

Tras esto está la palabra que identifica al objeto, es decir, un nombre que al ser dado en una orden Superglús sabrá de inmediato que nos estamos refiriendo a ese objeto, seguido por un adjetivo si procede, o el signo ‘_’ si el adjetivo es indiferente.

Por último tendremos los atributos de objeto, que básicamente vienen a ser propiedades que el objeto puede tener. Por el momento en Superglús sólo hay 3 atributos estándar, que son ‘da luz’, ‘es contenedor’ o ‘es prenda’, y corresponden a los atributos 0, 2 y 1 respectivamente.

Los atributos se numeran del 0 al 63 y como veis están divididos en dos bloques de 0s y 1s. Cuando hay un 0 significa que ese objeto no posee dicho atributo o cualidad, y cuando hay un 1 significa que sí.

Como veis se ha marcado el atributo de dar luz en la antorcha y el de prenda en el anorak.

En realidad la antorcha inicialmente esta apagada por lo que no da luz, así que cambiaremos el primer 1 de la definición de atributos de la antorcha por un 0.

```
/0  254    1    ANTORCHA     _     00000000000000000000000000000000 00000000000000000000000000000000
```

6. Vocabulario

El Vocabulario es una lista de las palabras que Superglús puede reconocer si el jugador las teclea durante el juego. Por lo tanto, cualquier palabra que no esté en esta tabla no tendrá ningún efecto.

Hemos dejado un vocabulario inicial que ya contiene los verbos, nombres, etc., más usados en las aventuras. Cada entrada para una palabra consiste en un máximo de diez letras que, o bien cubren la palabra completa, por ejemplo NORTE, o son las diez primeras letras de otra palabra más larga, por ejemplo *contabilizar*, seguido del valor para esa palabra y del tipo de palabra (por ejemplo: Nombre, Verbo, etc.).

Las palabras de igual tipo y numero en Superglús son consideradas sinónimas, y cualquiera podría usarse en lugar de la otra.

Todos los nombres con un valor menor de 50 son nombres propios, por ejemplo: nombres de personajes o lugares, pero más específicamente, para Superglús son nombres que no se pueden cambiar por la terminación verbal lo y la.

Explicándolo mejor: para mayor rapidez Superglús detecta que palabras son nombres propios (menos de 50) y cuales pertenecen a objetos manejables (mayores de 50).

En la frase "coge el martillo, golpea a Manolo en la cabeza y déjalo", si has dado a Manolo un valor menor de 50 y a martillo uno mayor, Superglús entiende que el la final del verbo se refiere al martillo.

Otro ejemplo: en la frase "dale a Carlos una tortilla y quítasela" entenderá; que el la final se refiere a la tortilla. El lo o la también puede ponerse antes del verbo con idénticos resultados.

Otro hecho importante es que las palabras con valor inferior a 20 son nombres, y que si Superglús no encuentra un verbo en la frase los convertirá temporalmente en verbos.

Por ejemplo: la palabra NORTE es un nombre y puede ser usado en la frase "ir al norte"; pero también puede ser tecleada sola, en cuyo caso como tiene un valor inferior a 20 se comportará como un verbo (o sea, que te vayas al norte).

Finalmente, las palabras con un valor menor de 14 se tomarán siempre como palabras-movimiento (cualquier palabra que sea una dirección) y sirven para determinar el mensaje que Superglús imprimirá si no se puede hacer nada con esa frase (por ejemplo, determinará si te responde un "no puedo" o "no puedo ir en esa dirección").

Hay que notar que esto de "menor de 14 es palabra-movimiento" se aplica tanto a verbos como a nombres convertibles en verbos.

Puesto que todos nuestros objetos son manejables, les daremos un valor mayor de 49.

ANTORCHA 50
BOLSA 51
BOCADILLO 52
MANZANA 53
BILLETE 54
PIEDRA 55
ANORAK 56

Accede a la solapa Vocabulario o a la sección VOC y escribe, detrás de donde pone ANTORCHA 50 noun:

BOLSA 51 noun

BOCADILLO 52 noun

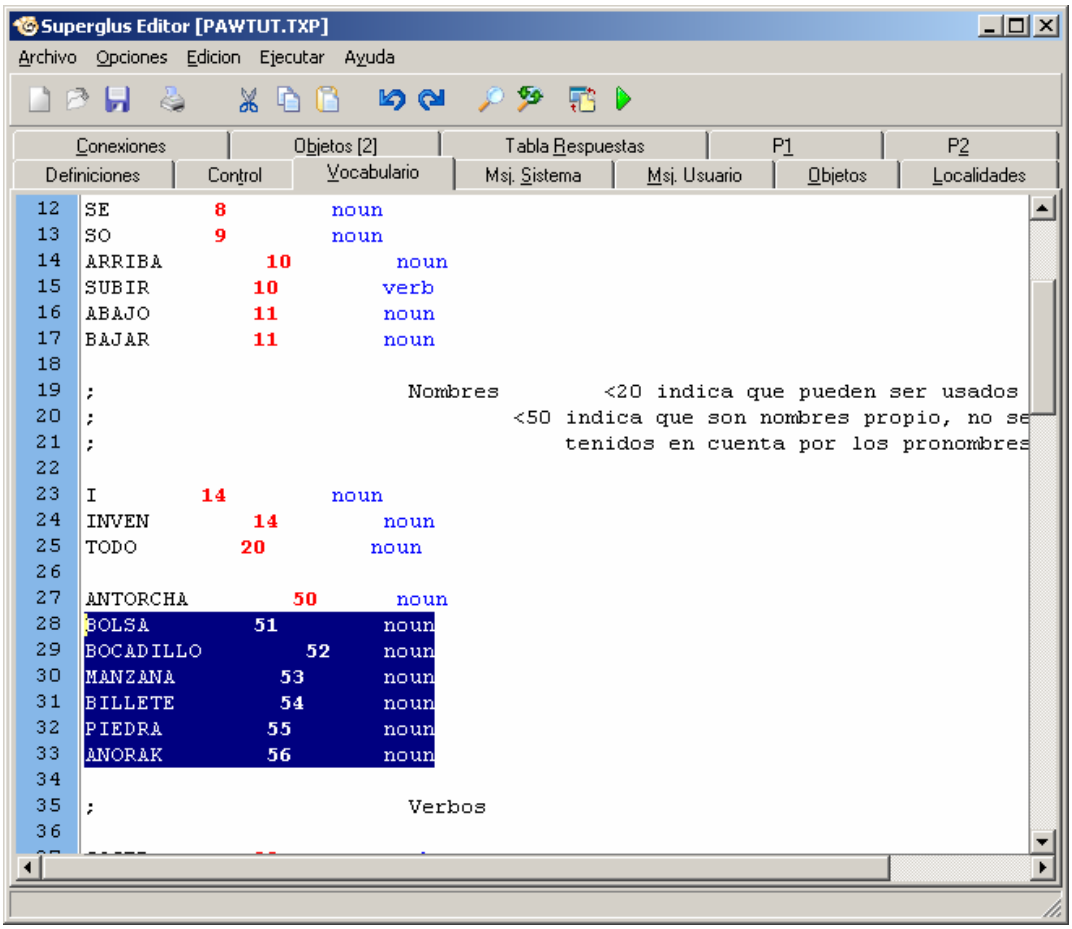
MANZANA 53 noun

BILLETE 54 noun

PIEDRA 55 noun

ANORAK 56 noun

Todos los números desde el 2 al 254, están disponibles para cualquier tipo de palabra y no hay limitación en el uso de sinónimos.
Si tratas de insertar una palabra que ya está presente, Superglús te lo avisará. Ten en cuenta que Superglús solo tomará las primeras 10 letras cuando te refieras a una palabra e ignorará el resto.



7. Flags/Banderas

Los flags o banderas son las variables del parser, claro que probablemente no sepáis que es una variable. Pues bien una variable es una especie de caja que contiene un papel con un número. Nosotros podemos comprobar el numero escrito en el papel, cambiar el papel por otro con otro número, sacar el papel y meter otro cuyo resultado sea multiplicar, sumar, etc. el numero que había dentro con otro, o con el papel de otra caja/flag, etc.

Hay 256 flags, numerados del 0 al 255, siendo algunos de ellos de uso interno de Superglús. En la guía técnica podéis ver un listado de dichos flags y su utilidad.

Normalmente los flags se utilizan para llevar el control de cosas que ocurren, para contar cosas, etc. Por ejemplo, pongamos que para cruzar un río antes debe salir el sol porque de otro modo el 'barquero' no quiere cruzarnos, pues bien, podríamos usar el flag nº 100 para saber en todo momento si hay sol o no, guardando por ejemplo un 1 en el flag 100 si hay sol y un 0 si no lo hay, de modo que podamos comprobar en todo momento si hay sol o no mirando lo que hay en dicho flag.

8. Diagnóstico (Debug)

Superglús incluye una orden llamada DEBUG, que puede tomar un valor de cero o cualquier otro distinto de cero. Si toma el valor de cero el sistema de debug queda desactivado, en caso contrario esta activado. Por defecto esta desactivado.

Así, si queremos tener habilitada la posibilidad de debug en una aventura deberemos incluir en cualquier sitio que seguro vaya a ejecutarse el condacto DEBUG 1, y a partir de ahí podrá accederse al debugger.

Un condacto se puede poner, por ejemplo, en la tabla de PROCESOS 1, que es una tabla que siempre se ejecuta tras cada descripción. Aprovecharemos para hacer una opción para que directamente se pase de la localidad 0 a la 2 al arrancar en el juego, nos vamos a la solapa P1 o bien a la sección /PRO 1 y escribimos al principio:

```
_       _       AT      0

                DEBUG 1

                GOTO  2

                DESC
```

Aquí vemos un primer ejemplo de programación en Superglús: Empezamos por dos guiones de subrayado, que indican que es indiferente que orden haya dado el jugador (el primero substituye al verbo y el segundo al nombre, al poner el _ indicamos que nos es indiferente.

Le sigue una condición, AT 0, que se cumplirá sólo si estamos en la localidad 0, y en caso contrario dejara de ejecutar esa *entrada*. Le sigue la orden DEBUG, que activa el debug. Tras eso, la orden GOTO 2 lleva al jugador a la localidad 2, y finalmente la orden DESC fuerza la descripción de dicha localidad (como si hubiéramos hecho 'mirar'). Si ejecutamos el juego observaremos que directamente aparecemos en la localidad 2, y lo malo es que no nos deja ver el texto de la localidad 0. Esto podemos solucionarlos poniendo una orden ANYKEY tras el

AT 0, porque ANYKEY es una orden que hace que el juego espere a que pulses una tecla. Quedaría por tanto:

```
_       _       AT      0

                ANYKEY

                DEBUG 1

                GOTO  2

                DESC
```

Es importante recordar quitar la orden DEBUG 1 de aquellas aventuras que vayan a ser preparadas para distribuirse al público, para que los jugadores no puedan hacer debug.

Una vez activado el debugger su uso es bastante sencillo, permitiendo ver y cambiar los flags de Superglús. Si desde la línea de comandos ponemos:

@flag (substituyendo la palabra flag por un numero de flag)

Superglús nos devolverá el valor actual de dicho flag, y no correrá turno.

Si ponemos:

@=valor (substituyendo valor por un numero)

Superglús asignará al último flag que hemos consultado dicho valor. Si no se hubiera consultado ningún flag anteriormente se tomará por defecto el flag 0.

Así, esta es una secuencia normal en un debug:

```
>@100

Flag 100 = 0

>@=20

Asignado valor.

>@100

Flag 100 = 20
```

Ahora si que seremos capaces de manipular los objetos en el juego. De momento la bolsa estará en la parada del autobús. Con nosotros llevaremos el bocadillo, la manzana, una antorcha apagada, y el anorak puesto.

Usa el diagnóstico para ver el valor del flag 1 (@1), y vemos que tiene el valor de 3, que es el número de objetos que se llevan en las manos, pero no puestos encima. Volvamos a la línea de INPUT y tecleamos "coger bolsa" y Superglús imprimirá el mensaje "Ahora cojo la bolsa".

Es lo que se llama AUTO-REPORTING (informe automático de cualquier acción que hayas hecho).

Esta última orden ha hecho que la posición de la bolsa haya cambiado de la localidad 2 que era la parada del autobús a la localidad 254 (que es una localidad especial para los objetos que pueden ser llevados pero no puestos).

Fíjate que no hay ningún cambio en la tabla donde se localizan los objetos al comienzo, Sino que lo que ha cambiado es la copia que se ha hecho de la base de datos cuando se empieza el juego.

Si ahora miras el valor de la bandera 1 verás que su valor ha aumentado a 4. Ahora puedes "quitar anorak" y recibirás el mensaje "No puedo quitarme el anorak, mis manos están llenas". Esto es porque Superglús inicialmente (si tu no le has dicho lo contrario) solo permite al jugador llevar cuatro objetos al mismo tiempo, y en algunos casos esta limitación puede impedirle al jugador quitarse vestido, etc. (de hecho, quitarse una prenda es cambiar la posición de ese objeto de la localidad 253, que es llevado encima, a la localidad 254 que es llevado en la mano).

Sí, por ejemplo, dejamos la bolsa primero y tecleamos "quitar anorak" veremos que sí se puede. Si miras otra vez la bandera 1 te darás cuenta de que todavía tiene el número 4; esto es porque al quitarte el anorak has aumentado el número de objetos que llevabas en tus manos.

Ahora como prueba teclea las siguientes órdenes y fíjate que hace cada una (usando las banderas):

Coger Bolsa
Quitar Anorak
Poner Anorak
Coger Manzana
Coger Billete de autobús

Hay que darse cuenta de que todas las respuestas, excepto la de la última orden, mencionan los objetos por su nombre; esto es porque estaban todos a la vista y por lo tanto el jugador sabía que existían (estaban en su propia localidad). Pero para un jugador que no conociera el juego, el billete todavía no existiría, y si se teclea "coger billete" ya en la respuesta lo mencionamos por su nombre, le estaríamos dando una pista muy importante.

Si tratas de poner cualquier cosa dentro de la bolsa descubrirás que Superglús lo que hace es dejar caer los objetos al suelo; esto es porque no le hemos dicho a Superglús todavía que cosas pueden ser puestas dentro de la bolsa.

Solamente le hemos dicho de momento que es un contenedor, en el próximo capítulo trataremos de este tema. Finalmente encontramos un terrible "error" en nuestro juego: si tecleamos "coger la puerta", se nos dará la respuesta de "No hay ninguna de esas aquí" o "No está eso aquí". No debería decir eso, porque en la descripción hemos dicho que hay una puerta.

Este es un problema muy frecuente, y ocurre porque le hemos dicho ha Superglús que hay una manzana, un bocadillo, etc. pero no lo hemos dicho que existe una puerta, y si usas coger o

dejar, etc. con cualquier palabra que no esté en el vocabulario entonces Superglús asume que es un objeto que "no está aquí". Sin embargo, una vez que esa palabra entre en el vocabulario, Superglús sabrá que no es un objeto (siempre que no haya una entrada para esa palabra en la tabla de relación objeto-palabras) y dará la respecto "no puedo hacer eso" que es la correcta.

Así que volvemos a la definición de vocabulario y añadiremos como nombres

puerta 57 noun
reja 58 noun
hierba 59 noun
camino 60 noun
banco 61 noun
estanque 62 noun
arbol 63 noun
rama 63 noun
hoja 63 noun

Es importante que Árbol, Rama y Hoja reciban el mismo número, porque no intentamos que sean manipulables, sino que solamente las entienda el PARSER. Si fueran manipulables, tendríamos que darles números diferentes. Esto es una consideración importante de diseño. Ahora puede ser el momento de hacer otro test del juego para ver que la orden "coger puerta" produzca una respuesta correcta. De momento hemos creado las localidades, las hemos conectado entre sí, hemos creado y descrito los objetos, les hemos asignado una palabra desde el vocabulario, un punto de comienzo en le juego, un peso relativo, si eran prendas, y si eran contenedores. El próximo capítulo tratará de como se crean algunos problemas y otros personajes para hacer que el juego sea más interesante.

8. Procesos y respuestas

La Tabla de Respuestas

La tabla de Respuestas es una forma especial de tabla de Procesos.

Una tabla de procesos puede verse como un lenguaje de programación secuencial (se ejecuta un comando cada turno). Las órdenes que se llevan a cabo se llaman "Condactos" porque se pueden dividir principalmente en dos grupos: condiciones y acciones.

Antes mencionamos que el PARSER de Superglús divide las sentencias en frases, las cuales son organizadas entonces en lo que conocemos como SL (sentencias lógicas).

En el caso de direcciones como NORTE (que es una sentencia lógica por si sola), Superglús usa la tabla de conexiones para descubrir hacia donde se ha de mover el jugador.

ANTES, sin embargo, Superglús mira en la tabla de RESPUESTAS para ver si hay alguna entrada en ella que se pueda asociar con la SL.

Es muy importante que quede claro el concepto de que cada frase que el jugador teclee y Superglús entienda, debe tener su correspondiente entrada en la tabla de Respuestas, excepto por la mayoría de los movimientos, que han sido ya puestos en la tabla de Conexiones.

La parte más importante de una SL es el verbo, que muestra la intención de la frase; la siguiente en importancia es el *primer nombre*, que muestra complemento directo de la SL. Por ejemplo, en "coger manzana", coger es el objetivo y manzana es el complemento directo.

Si miramos en el editor la tabla de respuestas, o bien el /PRO 0 si no usamos el editor veremos una entrada que pone:

I _ INVEN

Las dos palabras (porque _ es una palabra) indican el verbo y el nombre respectivamente. Como I es un nombre convertible en verbo (como vimos en la sección de Vocabulario) esto significa que si se teclea por si solo, es decir, si es la única palabra que ha tecleado el jugador en la frase, será tomada como un verbo en la SL. La línea (_) indica que el nombre no es importante en esta entrada, significaría como el "no palabra" que vimos en la tabla de objetos y palabras.

Lo que esto significa es, que si el jugador teclea I solo, Superglús lo equiparará con la primera entrada de la tabla de Respuestas que tenga la I, y ejecutará la orden que se describa al lado de ella.

Para ejecutar una sentencia, Superglús tiene que ejecutar cada uno de los condactos (órdenes) en una lista que ya tiene. Ahora viendo nuestro ejemplo la palabra I solamente tiene un condacto.

INVEN: es una acción (la parte activa de un condacto). Y es la parte activa porque se encarga de hacer una lista de los objetos que el jugador está llevando consigo o lleva puestos encima, en la pantalla. No te preocupes de momento de cómo hacer esto, solo nos interesa saber que lo hace.
Cuando tecleaste I (o INVENTARIO si lo has puesto como sinónimo) durante la prueba del juego, fue esta entrada en la tabla de Respuestas la que causó que pasase algo, porque se creó por el PARSER una sentencia lógica de "I _", que entonces Superglús emparejó con la primera entrada en la tabla-respuesta.

INVEN, después de que ha hecho una lista de todos los objetos que lleves, ya le dice automáticamente a Superglús que ha hecho algo, y cuando Superglús descubre esto, busca en el PARSER otra sentencia lógica, la cual el PARSER ofrece, decodificando las siguientes frases en el INPUT del jugador. Superglús entonces coge esta nueva SL y la compara con todas las entradas que hay en la tabla de Respuestas, y así sucesivamente.

Este bucle (loop) se muestra en el diagrama 4 en forma de una carta de flujo que se puede seguir desde el cuadrado marcado con "comienzo". El bucle es bastante más complejo que lo que el diagrama te pueda parecer y de hecho se da una versión más completa en la guía técnica, pero de momento el diagrama 4 nos servirá.

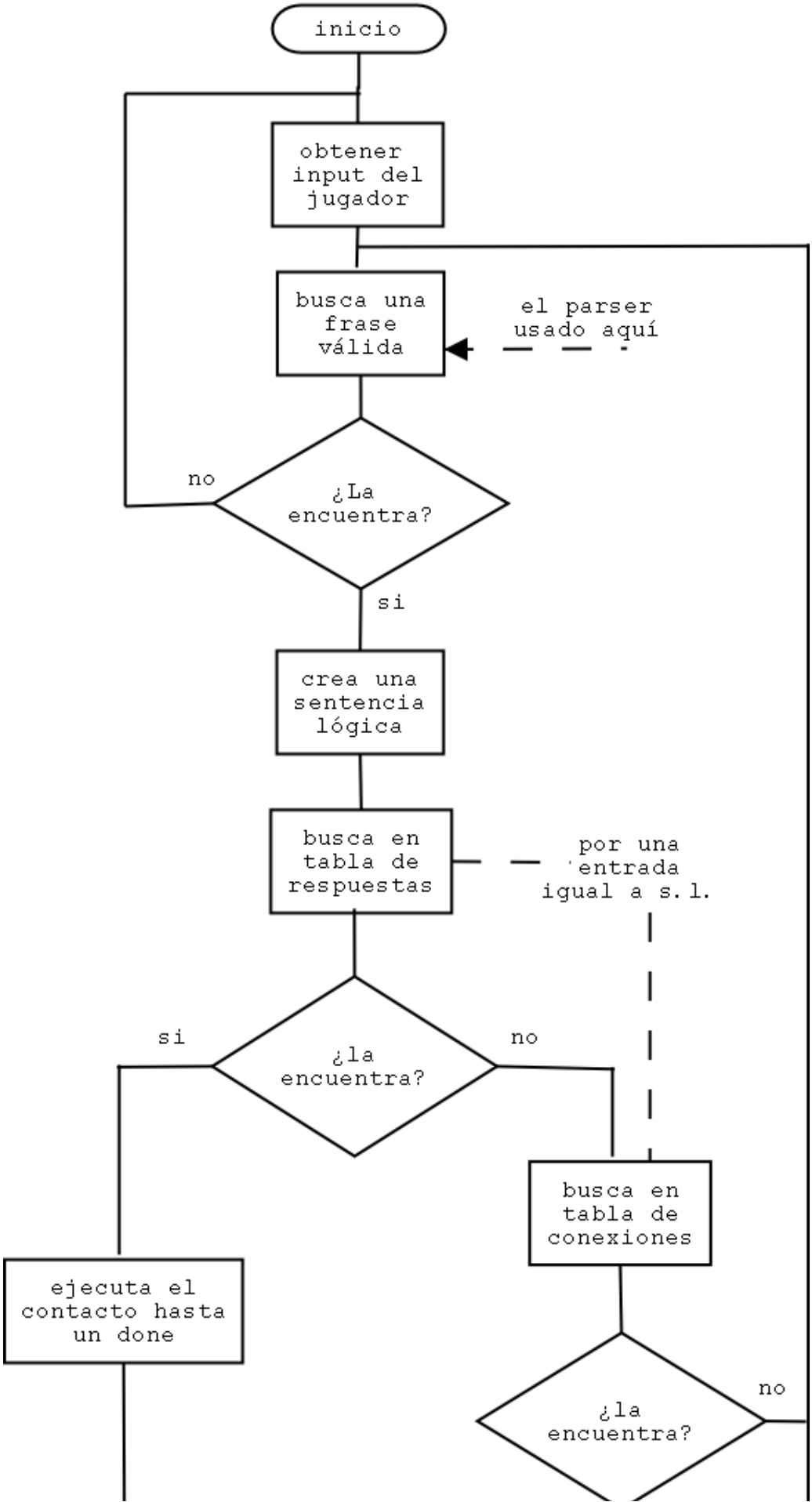

inicio
obtener input del jugador
busca una frase válida
el parser usado aquí
no
¿La encuentra?
si
crea una sentencia lógica
busca en tabla de respuestas
por una entrada igual a s.l.
si
¿la encuentra?
no
ejecuta el contacto hasta un done
busca en tabla de conexiones
no
¿la encuentra?

Diagrama 4

Te aconsejamos fervientemente que vuelvas a leer los párrafos anteriores y estudies el diagrama, hasta que entiendas como Superglús trabaja sobre una sentencia lógica antes de funcionar.

Vamos a considerar la otra entrada de Respuestas que se encuentra algo mas abajo:

QUIT _ QUIT
TURNS
END

Bien, QUIT es un verbo en el vocabulario, así que como la mínima frase que Superglús considerará válida es un verbo, si QUIT se teclea solo entonces ya el PARSER generará una SL de "QUIT _".

En su búsqueda a través de la tabla de Respuestas, Superglús encontrará la entrada que hemos visto más arriba y empezará a ejecutar los condactos.
El condacto QUIT, que casualmente coincide con la palabra pero no tiene nada que ver es una condición (la parte condicionante de la palabra condacto). Una condición solo decide si Superglús continúa para llevar a cabo el siguiente condacto de la lista, es decir, no toma de por sí ninguna acción, sino una decisión.

QUIT lo que hace es determinar si el siguiente condacto puede ser ejecutado, preguntándole al jugador "¿Estás seguro?". Si el contesta NO; entonces QUIT avisa a Superglús que ha hecho "algo" y este irá en busca de otra SL (dejando en paz QUIT).

Esta es una forma un poco diferente de tratar las condiciones en Superglús, pero QUIT es una condición bastante especial, como veréis en el futuro. Si el jugador teclea "SI" entonces QUIT no hace nada, y deja que Superglús busque el siguiente condacto de la sentencia, que será entonces TURNS.

TURNS es una acción que imprime "has hecho x ordenes" o "has dado x órdenes" en la pantalla, donde x es el número de frases que el jugador ha tecleado desde el comienzo del juego. Aparte de este hecho, como tú no le has dicho a Superglús que deje de buscar condactos, entonces éste continuará buscando END.

END es una acción también especial que imprime "¿apetece otro juego?", o "¿quieres jugar otra partida?" en la pantalla. Si el jugador teclea "SI", entonces END reiniciará el juego con todos los objetos en sus posiciones iniciales, etc. Si el jugador dice "NO" entonces generará un error "OK", lo cual le hará salir del juego.

Las otras entradas que ya están presentes en la base de datos, se hacen cargo de varios comandos estandarizados que los jugadores normalmente necesitarán en la aventura.

Los condactos usados en las otras entradas se discuten más abajo.
Te estarás preguntando por qué estas entradas están en una tabla y no forman parte ya de por sí de Superglús, si son necesarias en cada juego.

Pues bien, la razón es que, aparte del hecho de que es mucho más fácil ponerlas en una tabla que en el propio juego, **TU JUEGO A LO MEJOR NO LAS NECESITA**, y de esta forma las puedes borrar cuando quieras.

DESC es una acción, como se comento en un capítulo anterior. De hecho, es la que se usa en la entrada "M _" de la tabla. Y esta acción hace que Superglús abandone la búsqueda por la tabla de respuestas y reDEScriba la localidad actual del jugador. También tiene varios sinónimos: mirar, describir, etc. en el Vocabulario.

SAVE y **LOAD** son dos acciones que permiten hacer un SAVE y un RELOAD del estado actual del juego en cinta. La posición actual del juego incluye también cualquier otra pieza de información que se necesite para restaurar exactamente el juego en el mismo estado en que estaba; ello incluye los valores de las banderas, las posiciones de los objetos, y mucha más información.

No debes confundir aquí los anglicismos SAVE y LOAD que hemos dejado en el vocabulario con las acciones SAVE y LOAD. Nosotros hemos incluido también otros sinónimos, como GRABAR y CARGAR, en el Vocabulario, pero todos ellos usarán las acciones SAVE y LOAD de la tabla de Respuestas.

Es de notar que ambos SAVE y LOAD ya de por sí hacen una acción DESC cuando han terminado. Lo cual significa que cualquier condacto que siga, será ignorado igualmente por el INPUT del jugador.

RAMSAVE y **RAMLOAD** son dos acciones similares a SAVE y LOAD, excepto que ellos usan un "buffer" (área de memoria libre) para guardar la posición del juego. En realidad se usan por compatibilidad con el Professional Adventure Writing System de Spectrum, y no porque tengan sentido hoy en día.

Si a Superglús se le acaban los condactos en una lista sin haberle aclarado que ya se ha hecho (done) algo, sencillamente "caerá" hasta el final, y al darse cuenta de esto, continuará en su búsqueda de otro SL.
También dijimos que QUIT era una condición bastante especial. Para comenzar, es la única condición que busca información del jugador, y por otra parte, también le informa a Superglús de que algo ha sido hecho si el jugador teclea "NO" (es decir, que no quiere abandonar el juego), lo cual hace que Superglús comience a buscar otro nuevo SL.

MUY IMPORTANTE: una condición normal, si "falla", sencillamente hará que Superglús continúe buscando en la tabla de respuestas otra condición que haga juego con la SL.

Los otros condactos que está usados se considerarán ahora en relación con las entradas de las cuales son parte. Para simplificar nuestra explicación, debemos considerar la posición de un objeto en el juego.

Puede estar en cuatro lugares:

- AQUI: es la localidad actual del jugador (es el valor que se guardaba en la bandera 38, si recuerdas bien). Tiene el valor de 255, HERE o @38 (pueden usarse cualquiera de esas tres expresiones en la programación para indicar la localidad actual, mas explicaciones en la guía técnica.
-
- LLEVADO PERO NO PUESTO: Localidad 254, que es una localidad imaginaria donde se ponen todos los objetos que el jugador lleve.
-
- PUESTOS ENCIMA: Localidad 253, que es una localidad imaginaria donde todos los objetos que el jugador lleve puestos encima son guardados.
- NO PRESENTES: ¡En cualquier otro sitio! Esto también incluirá la localidad 252, que es una localidad imaginaria donde todos los objetos que aún no existen son guardados.

Vamos a ver otras dos entradas de la tabla de Respuestas:

COGE TODO DOALL 255 (Parámetros: Localidad donde estás: Aquí,)

COGER _ AUTOG
DONE

Estas dos entradas son las que permiten al jugador coger cualquier objeto. Coger un objeto significa cambiar su localidad desde AQUI (255) a LLEVADO(254). Ignoremos por ahora la entrada "coge todo" y vamos a estudiar la orden "COGER _". Como dijimos antes, esa línea baja significa "cualquier palabra", así que no importa cual nombre teclee el jugador en una frase que contenga el COGER. La entrada COGER _ siempre casará (esto es lo que se llama "cargando" o "preparando" una entrada). Veamos la frase "coger la manzana": *la* será ignorada por Superglús porque no está en el Vocabulario, pero la SL será "coger manzana", y esto "cargará" la entrada "COGER _" cuando Superglús encuentre este condacto.

AUTOG es una acción que AUTOmáticamente cogerá cualquier objeto que haya sido especificado por el nombre. Es donde la tabla de Palabras-Relacionadas con Objetos viene a funcionar. AUTOG mira a través de esa tabla buscando una entrada que sea similar al nombre de la SL cuando la encuentre (manzana en nuestro ejemplo) entonces sabe el número del objeto al cual se refiere (en el caso de la manzana es 3).

Se asegura entonces de que la localidad actual de ese objeto sea AQUI que es (255) y la pasa a LLEVADA que es (254) y entonces imprime el mensaje "Ahora tengo el objeto _." donde la línea baja es reemplazada por la descripción del objeto actual, por ejemplo el que acaba de marcar el AUTOG.
Si no encuentra ninguna entrada, entonces hay cinco posibilidades:

1. El jugador ha intentado coger un objeto que ya lleva o tiene puesto, en cuyo caso el mensaje "Ya tengo el _" se imprime.
2. El jugador ha intentado coger un objeto que está en la localidad AQUI NO, en cuyo caso se imprimirá el mensaje "Aquí no hay uno de esos".
3. El jugador ha intentado coger algo que no es un objeto, pero que tiene una palabra en la tabla de Vocabulario (por ejemplo: puerta en nuestro juego de demostración). Esto producirá el mensaje "No puedo hacer eso".
4. El jugador ha usado una palabra que no está en el Vocabulario. Eso hace que el PARSER cree una SL de "COGER _" lo cual "carga" nuestra entrada de "COGER _" de todos modos. AUTOG asume que se trata de un nombre que describe un objeto (que puede o no existir) y entonces imprime el mensaje "No hay uno de esos aquí"
5. El jugador no puede llevar más objetos, o el peso de este objeto sobrepasa el límite permitido paa el jugador, en cuyo caso aparecerá un mensaje que le informa de ello.

Si el AUTOG funciona, entonces Superglús mira el siguiente condacto.

DONE solamente le dice a Superglús que la entrada ha terminado y que debe ir y buscar otra SL.

Ahora miraremos la entrada *COGER TODO*. Como habrás supuesto, esta entrada lo que hace es intentar coger todos los objetos que estén en la localidad donde estás tú.

Si el jugador teclea la frase COGER TODO, el PARSER creará una sentencia lógica (SL) de "COGER TODO" y entonces hará pareja con la entrada presente en la tabla y Superglús pasará a ejecutar la acción DOALL.

DOALL es una acción que debe ser seguida con un parámetro. El parámetro nos da el número de la localidad que usaremos.

Lo que hace DOALL es buscar la lista de localidades en la cual se encuentra cada objeto, intentando encontrar entradas que sean iguales para el parámetro que se ha dado (que en este caso es 255, una localidad especial que significa que se use la localidad en la cual el jugado esté presente).

Cuando encuentra uno similar, o el mismo, entonces se va a la tabla objetos relacionados con palabras para encontrar la palabra del Vocabulario que describe el número del objeto.

Esta palabra se pone en la actual SL (reemplazando el nombre todo) y una bandera interna se activa para indicar que DOALL está activo.

Luego Superglús sigue mirando el resto de la tabla de Respuestas buscando una entrada que haga juego con la SL recientemente modificada.

Esta entrada será COGE _ (de la cual hemos hablado antes), que cogerá el objeto que Superglús haya buscado en la tabla de Objetos.

Cuando haya terminado de hacer lo anterior, Superglús se dará cuenta de que todavía permanece activo el DOALL, o sea, la bandera está activa y vuelve otra vez a la entrada COGE TODO.

En realidad lo que hace es saltar directamente a la acción DOALL y buscar otro objeto para generar una nueva SL, y así sucesivamente para todos los objetos que estén en la localidad especificada.

Cuando se acaban los objetos, entonces la bandera se desactiva para indicarle a Superglús que no está activo el bucle y se le dice que busque otra SL.

Esto podría parecer un modo bastante complejo de ejecutar esa acción, pero si examinamos a las entradas similares que tratan de PONERSE, QUITARSE y DEJAR, veremos que utilizan el mismo mecanismo: AUTOD, AUTOW y AUTOR trabajan de una manera muy similar a AUTOG, mientras que DOALL sencillamente se limita a cambiar la localidad actual para ejecutar las acciones.

Por ejemplo, usa 254 (LLEVADAS) como el parámetro en la acción DROP y también en la acción WEAR (es decir, DOALL busca todos los objetos que lleves cuando intentas dejarlos caer o ponértelos encima); y utilizará la localidad 253 (LLEVADO PUESTO ENCIMA) cuando se trata de quitarse todo. Si lo anterior te parece un poco complicado, no te preocupes de momento, porque DOALL es uno de los condactos más complejos de Superglús y poco a poco lo irás entendiendo.

Sería interesante que en este momento hagas una prueba de la aventura, y trates todos los comandos de "dejar todo", "coger todo", etc. para que el mecanismo se te haga más claro.

9. Personajes

El pajarito

El juego de práctica que hemos hecho es bastante fácil de resolver. Así que le vamos a añadir

un poco de complejidad creando dos caracteres que vayan vagando por nuestro pequeño mundo. Estos caracteres son llamados PSI, o Caracteres Pseudo Inteligentes, porque es obvio que aunque no pueden pensar, debe de parecerle al jugador como si lo hiciesen.

Un PSI consiste principalmente en una colección de mensajes, banderas y entradas en tablas de procesos. Verás que con unas pocas entradas simples se pueden crear efectos sorprendentemente reales.

La creación de un complejo PSI puede tomar mucho más tiempo, pero en general sigue los mismos principios que los que vamos a hacer con los nuestros: un pajarito y un perrito.

El pájaro se pone para complicar el escenario de la siguiente manera: cogerá el billete al principio del juego (normalmente se le daría una localidad no usada como contenedor de los objetos que lleve el pájaro, pero nosotros usaremos la localidad 252, puesto que solamente tenemos un PSI que puede tener un objeto).

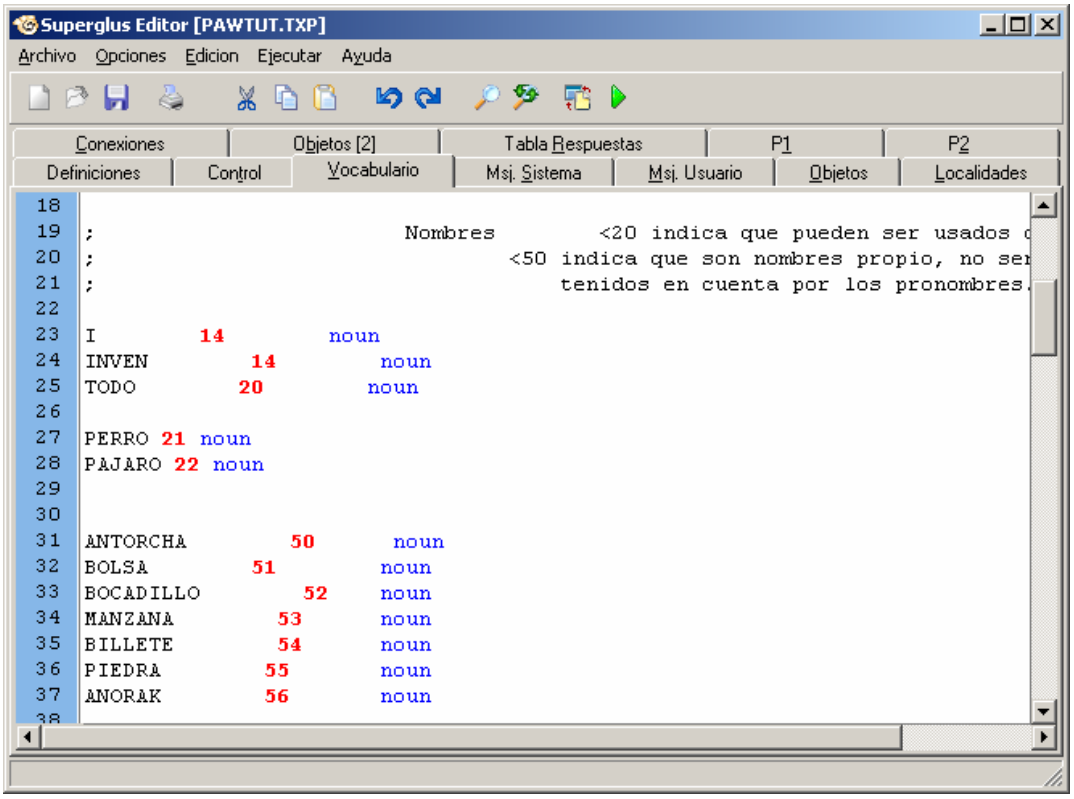
De modo que el jugador debe convencer al pajarito para que deje caer el billete. El intentar quitárselo o cogerlo dará el mensaje "No puedo hacer eso", y hará que el pajarito se vaya volando hacia otra parte.

El pajarito también dará la paliza volando entre el pabellón de música y una ramita del árbol, y esto lo hará a intervalos regulares. La única manera de convencer al pajarito para que deje caer el billete de autobús es dejar caer el bocadillo en la misma localidad.

Lo primero que hay que hacer es ir a los objetos y cambiar la localidad inicial del objeto 4 a no cerado (252) Esto hace que el objeto no exista como objeto, porque lo lleva el pajarito en el pico.

En el vocabulario insertamos también los nombres pajarito o pájaro y perro.

Supondremos que el pájaro es 22 y el perro es 21, perro=21, pájaro=22,



Después, insertamos los siguientes mensajes.

/6
El pajarito deja caer el billete para coger el emparedado
/7
El pajarito se lleva velozmente el billete
/8
El pajarito me ignora
/9
Un pajarito anda rondando por aquí
/10
El pajarito tiene un billete en su pico
/11
Un pequeño pajarito se posa en la hierba
/ 12
El pajarito está ahora en la rama del árbol
/13
El pajarito ve al perro y sale pitando
/14
El pajarito se va

Esto se hace en la solapa ‘mensajes de usuario’ (cuidado, no ‘mensajes del sistema’) on el apartado /MTX.

Es importante no dejar líneas en blanco entre mensajes porque serán consideradas como parte del mensaje y escritas en el juego.

Luego pondremos los mensajes que vaya a usar el perro, pero ahora debemos seleccionar la tabla de Procesos y prepararnos para otras de las virguerías de Superglús.

Sub-procesos

En una aventura de verdad que contenga varios PSI y un montón de acciones, las tablas de procesos 1 y 2 pronto acabarán llenas, y se hace cada vez más difícil trabajar de esta forma. Entonces viene el momento de dejar que las otras tablas de Procesos comiencen a actuar. Estas pueden ser llamadas desde las tablas de Procesos 1 y 2 o de la tabla de Respuestas, y usadas como una extensión de la tabla desde la cual fueron llamadas.

El llamar a otro proceso hace que Superglús guarde en memoria dónde estaba en el momento de ser llamado, y continuará actuando en la misma forma en que lo estaba haciendo. Por ejemplo, si se llamó desde una tabla de Respuestas, Superglús tratará de hacer coincidir la Sentencia Lógica actual con cualquier otra entrada. Pero si es llamado desde una tabla de Procesos, sea la 1 o la 2, Superglús sencillamente ejecutará cada entrada (ESTO ES MUY IMPORTANTE).

Por otra parte, cuando algo es DONE en el Proceso llamado, Superglús volverá a la tabla original y continuará trabajando. Por lo tanto se pueden hacer acciones bastante complejas mediante el uso de los sub-procesos. Los usuarios que sepan algo de programación reconocerá fácilmente que son sencillamente sub-rutinas.

Cuando Superglús está en un sub-proceso, es posible que sea llamado para actuar en otro sub-proceso (¿un sub-sub-proceso?), y así se puede continuar con sub-sub-sub hasta un valor de 10.

En este momento no vamos a usar un sub-proceso muy complicado, solamente uno simple, y lo haremos para tener en cuenta todas las chorraditas del pájaro.

De momento deberíamos estar comenzando el proceso 3. Para crearlo podemos usar la opción del menú Archivo del editor, o simplemente añadir al final del fichero SCE si usamos un editor externo : /PRO 3

Como solamente hay unas pocas entradas en la tabla, usaremos el mismo par de palabras ("_ PAJARO"), aun cuando hay que tener en cuenta que al escribir tu juego debes de usar otras palabras que te recuerden lo que hace cada entrada.

El flag 111, será una bandera de "trabajo" (auxiliar), porque contiene un valor que se usa solamente como comparación.

El flag 112 llevará el número de la localidad actual del pájaro.

La bandera 5 es una bandera especial, porque si tiene un valor diferente de 0, disminuirá 1 cada vez que Superglús ejecute una acción, es decir, cada vez que Superglús revise la tabla de Procesos 2. Es lo que se llama una bandera auto-decreciente.

En este caso usaremos la bandera 5 para contar el número de turnos que han pasado en el juego. Un turno del juego es toda una vuelta del circuito grande en el diagrama 4, y de momento, esto pasa cada vez que el jugador teclea una frase.

El pajarito cambiará de localidad cada tres frases, lo que creará una especie de apariencia de acción independiente de lo que teclee el jugador.

Ahora vamos a insertar las siguientes entradas (sin los comentarios, como hicimos antes). En cada entrada tenemos una explicación de su propósito y también explicaremos cada nuevo condacto que se use:

Primero hay que determinar si el pájaro va a salir pitando. Esto pasará cuando la bandera 5 tenga el valor 0 (empezará a contar desde 3). Entonces, si el billete está en la misma localidad que la del pájaro, será destruido (Puesto que la localidad 252 indica que el pájaro lo tiene) y si el jugador está en la misma localidad que el pájaro, se le dirá que el pájaro se ha llevado el billete.
Tienes que darte cuenta de que el pájaro sigue con sus ciclos de movimientos aunque el jugador no lo vea. De hecho, en el mundo de Superglús un árbol se cae aunque no haya nadie para verlo. Así de real es.

```
_ PAJARO COPYOF   4     111 ;Copia la localidad del objeto 4 (el billete) a la bandera 11
          SAME    111 112 ;y mira si está en la misma localidad que el pájaro
          ZERO    5       ;¿Va a volar el pájaro?
          DESTROY 4       ;El pájaro coge el billete
          SAME    112 38  ;¿Está el pájaro en la misma localidad que el jugador?
          MESSAGE 7       Díselo al jugador
```

Fíjate, no hay acción DONE porque queremos que Superglús haga cada entrada, una detrás de otra, siempre que se cumplan los requisitos de las condiciones de ellas. La tabla anterior te muestra como se pueden mezclar condiciones y acciones para crear nuevas condiciones.

COPYOF: Es una acción que debe ser seguida por el número de un objeto y una bandera. Lo que hace es que copia la localidad actual del objeto que se especifique, en la bandera especifica. La usamos en esta situación para ver si el billete está en la misma localidad que el pájaro.

SAME: Es una condición que compara el contenido de dos banderas, y que es favorable o positiva si ambas son del mismo valor.

DESTROY: Es una acción que pone un objeto especificado en la localidad 252 (localidad no creada).

Ahora vamos a crear dos posibles movimientos para el pájaro. Si el pájaro está en el pabellón de música y el flag 5 ha llegado a 0, entonces hay que mover el pájaro, igualar el flag 5 a tres otra vez y decirle al jugador que el pájaro se ha ido, si está en la misma localidad. Y lo mismo si el pájaro está en la rama.

```
_ PAJARO EQ      112 8 ;¿Está el pájaro en la rama?
          ZERO    5     ;¿Es tiempo de volar?
          LET     112 5 ;Mover el pájaro al pabellón
          LET     5   3 ;Tres frases antes de moverse
          AT      8     ;¿Está el jugador también aquí?
          MESSAGE 14    Dile que el pájaro ha volado


_ PAJARO EQ      112 5 ;¿Está el pájaro en el pabellón?
          ZERO    5     ;¿Es tiempo de volar?
```

```
            LET     112 8 ;Mover el pájaro a la rama
            LET     5   3 ;Tres frases antes de moverse
            AT      5     ;¿Está el jugador también aquí?
            MESSAGE 14    Dile que el pájaro ha volado
```

EQ: Es una condición que debe ser seguida por un número de bandera y otro valor, y que será positiva si la bandera contiene ese valor. En este caso (EQ 12 8) está comprobando si el pájaro está en una localidad especificada.

LET: Es una acción que también es seguida por una bandera y un valor. Y pone ese valor en la bandera.

Ya hemos tenido en cuenta el vuelo del pájaro. Ahora vamos a poner sus llegadas, y si llega a una localidad donde está el jugador, hay que decírselo.

```
_ PAJARO EQ      5   3  ;¿Acaba de volar el pájaro
          SAME    112 38 ¿Está en la localidad del jugador?
          AT      5      ;¿o en el pabellón de música?
          MESSAGE 11     ;Ha aterrizado en la hierba

_ PAJARO EQ      5   3  ;¿Acaba de volar el pájaro
          SAME    112 38 ¿Está en la localidad del jugador?
          AT      8      ;¿o en la rama?
          MESSAGE 11     ;Ha aterrizado en la rama
```

Ahora, si el pájaro tiene el billete en su pico debemos decírselo al jugador.

```
_ PAJARO EQ      5   3
          SAME    112 38
          ISAT    4   252 ;¿El billete no ha sido creado?
          MESSAGE 10      ;Tiene el billete en su pico
```

ISAT: Es una condición seguida de un objeto y el número de una localidad y es positiva si el objeto está en la localidad especificada.

Finalmente, si el bocadillo está en la misma localidad que el pájaro, el pájaro dejará el billete para cogerlo. Esta entrada no tiene nada que ver con la bandera 5, así que será buscada cada vez que Superglús vaya al Proceso 2.

Por lo tanto, si el jugador deja caer el bocadillo <u>después</u> de que el pájaro haya llegado, la secuencia correcta aun se llevará a cabo.

```
_ PAJARO COPYOF  2   111 ;emparedado
          SAME    111 112 ;¿En la misma localidad que el pájaro?
          ISAT    4   252 ;¿Lleva el billete en el pico?
          COPYFO  112 4   ;Empieza a caer el billete
          SAME    112 38  ;¿Está el jugador también por allí?
          MESSAGE     6   ;Díselo
```

COPYFO: Es una acción que copia el contenido de una bandera específica a la localidad actual del objeto específico. También hay acciones COPYFF y COPYOO que son fáciles de comprender.

Así completamos las rutinas de control del pajarito. Pero necesitamos una entrada en el Proceso 2 para llamar a todas esas rutinas cada ciclo, así que busquemos en la tabla 2 y pongamos la entrada que llama a todo el subproceso, es decir, la entrada "madre".

```
_ PAJARO PROCESS 3
```

lo que hará que Superglús ejecute nuestros controles de pájaro cada vez que pase por ahí.

Ahora debemos asegurarnos de que el pájaro empieza en la localidad correcta y que el jugador sabe que el pájaro está ahí cuando la localidad se describe (o empezará a ver mensajes acerca de un pájaro dando la paliza sin haber tenido ninguna descripción de que existe tal pájaro).

Así que seleccionamos el Proceso 1 (que ya sabemos que se llama después de cada descripción de localidad) y vamos a corregir la entrada existente con _ _ poniéndole un [LET 12 8], lo cual hará que el &pájaro esté en la rama al principio del juego. La entrada modificada debe de ser:

```
* * AT        0
   ANYKEY

   DEBUG 1

   LET       112 8 ;El pájaro está en la rama (localidad 8)

   GOTO      2
   DESC
```

Y también hay que poner la siguiente entrada en la tabla de Procesos, para que le diga al jugador que hay un pajarito, y que tiene el billete.

```
_ PAJARO SAME       112 38
         MESSAGE 9
         ISAT       4    252
         MESSAGE 10
```

Por último seleccionaremos la tabla de Respuestas e insertaremos la entrada:

```
COGER BILLETE SAME      112 38  ;¿Está el pájaro en la misma localidad?
              ISAT      4   252 ;¿Con el billete en el pico?
              CLEAR     5       ;Esto lo fuerza a volar
              NOTDONE           ;"No puedo hacer eso"
```

Esta entrada se disparará antes de la entrada COGER _ y previene la respuesta "No hay uno de esos aquí" si el pájaro está presente con el billete.

CLEAR: Es una acción que va seguida por un número de bandera, y que pone esa bandera a 0. En este caso hará, que el pájaro salga pitando, simulando su miedo a una gran mano que desciende sobre él para coger su preciada posesión.

NOTDONE: Es una acción similar a DONE, pero que engaña a Superglús pensando que no has hecho nada y por lo tanto hace imprimir el mensaje "No puedo hacer eso".

Y ahora, vamos al momento de la verdad. Hay que probar el juego para ver si el pájaro vuela del pabellón de música a la rama. Juega un rato para ver si el pájaro continúa con su existencia vagabunda. Después trata de dejar caer el bocadillo en la misma localidad. Date cuenta de que si no coges el billete antes de que el pájaro se pire, el pájaro lo cogerá otra vez.

Y menos mal que acabamos con el bendito pájaro. Es complicado, pero te enseñará un montón sobre Superglús.

El perro

Este perro lo ponemos para complicar el juego un poco más. Simplemente seguirá al jugador donde quiera que vaya y espantará al pajarito. Como no se espera que el perro se suba al árbol, debemos impedir que el jugador pueda tentar al pájaro con el emparedado desde la rama. Para hacer eso disponemos que cualquier objeto que se suelte desde la rama del árbol caiga hasta el suelo. El jugador se puede desembarazar del maldito perro poniéndole la cadena y amarrándola al banco. Además, el jugador puede "hablar" al perro, lo cual le dará otro medio para desembarazarse de él, diciéndole que se SIENTE o que se QUEDE.

Antes de examinar las entradas en las tablas de Procesos y Respuestas., necesitamos tener el control del perro por medio de las siguientes palabras en el vocabulario y mensajes en la tabla de mensajes.

Verbos		Nombres	
ATAR	34		
DESATAR	35		
SENTAR	36		
QUEDAR	36		
VEN	37	AQUÍ	37

Los números te demuestran que sentarse y quedarse son lo mismo, y que también usamos un Nombre y un Verbo con el mismo número.

Mensaje 15

El _ cae al suelo al pie del árbol

El que pongamos un guión bajo en él tiene un propósito especial, del cual ya hablaremos más adelante.

Mensaje 16

El perro me mira con amor

Mensaje 17

El perro está aquí

Mensaje 18

El perro me sigue moviendo la cola

Mensaje 19

El perro va arrastrando la correa

Mensaje 20

El perro está atado al banco con una cadena

Mensaje 21

Confiadamente, el perro me deja ponerle la correa alrededor del cuello

Mensaje 22

He amarrado el perro al banco

Mensaje 23

¿A quién se lo digo?

Mensaje 24

El perro se sienta tranquilo

Mensaje 25

He desatado al perro del banco

No hay verdadera necesidad de hacer la rutina para el perro en una tabla de Procesos separada, porque solamente hay una entrada, pero lo haremos por si quieres ampliar el juego más tarde.

La bandera 113 contendrá la localidad actual del perro.

La bandera 114 contendrá varios indicadores:

  0 si el perro está libre para andar por ahí
  1 si el perro tiene la correa alrededor del cuello
  2 si el perro está atado al banco
255 si el perro está sentado muy tranquilo

Lo primero creamos una tabla de procesos, la 4.

```
_ PERRO NOTSAME 112 38   ¿No está el perro donde está el jugador?
        LT      114 2    ;¿Todavía se puede mover?
        NOTAT   8        ;¿El jugador no está arriba del árbol?
        COPYFF  38  113  ;Mueve el perro a la localidad del jugador
        MESSAGE 18       ;y dile que lo atosigue
```

Suponemos que eres capaz de entender qué es NOTSAME, NOTAT y COPYFF, en caso negativo, en la guía técnica te explicaremos para qué sirve cada una.

LT: Es una condición que es positiva o favorable si la bandera especificada contiene un valor MENOR QUE el valor especificado. En este caso [LT 14 2] solamente será positivo o acertado si la bandera 14 tiene un valor menos de 2.
Ahora vamos a la tabla de Procesos 2:

```
_ PERRO PROCESS 4
```

es decir, que estás usando la tabla de Procesos 2 para enviar a Superglús a la tabla de Procesos 4.

La entrada "_ Perro", debe estar antes de la entrada del pajarito. Es para asegurar que el perro se mueva a la nueva localidad del jugador antes de que el pájaro sea comprobado.

Similarmente a como hicimos con el pájaro, se requieren entradas en la tabla de Procesos 1, para informarle al jugador de que el perro anda por ahí rondando:

```
_ PERRO SAME      113 38 ;¿El perro está en la misma localidad?
        MESSAGE 17       ;Díselo al jugador
        EQ        114 1  ;¿Con la correa?
        MESSAGE 19       ;Sí díselo al jugador

_ PERRO SAME      113 38
        EQ        114 2  ;¿Está el perro amarrado al banco?
        MESSAGE 20

_ PERRO SAME      113 38
        GT        114 2  ;255 es mayor que 2, así que
        MESSAGE 24       ;Dile al jugador que el perro está sentado
```

Ya que estamos en la tabla de Procesos 1, modifiquemos la entrada * para que también tenga [LET 13 2] (antes del GOTO), esto hace que el perro empiece en la parada del autobús.

Ahora, para que el perro asuste al pajarito, necesitamos una entrada extra en la tabla de Procesos 3. Pero debe ir delante de la entrada que decide que el pájaro deje caer el billete y después de la entrada que hace que el pájaro vuele. Así nos aseguramos que el pájaro volará con el billete si lo tiene y lo deja si no lo tiene.

Por lo tanto, necesitamos insertar otra entrada antes de la sexta. Usaremos [I _ PAJARO 6] para hacerlo.

```
_ PAJARO SAME     112 113 ;El pájaro y el perro están en la misma localidad
          LET     112 8   ;Sólo en el pabellón de música
          LET     5   3   ;Muévelo a la rama, espera 3 frases
          AT      5       ;¿Está el jugador en el pabellón de música?
          MESSAGE 13      ;Dile que el pájaro se ha ido
```

El último cambio en la tabla de Procesos es insertar un sub-proceso que luego llamaremos desde la tabla de Respuestas para que se haga cargo de hablar al perro. El mecanismo es muy simple. Si el jugador incluye una frase "(" ")" en su sentencia de INPUT, entonces el parser guardará memoria del lugar donde esté y seguirá descifrando la frase entre comillas.

Hay una acción llamada PARSE que instruye a Superglús para que decodifique la cadena de palabras que el jugador haya tecleado, haciendo entonces la Sentencia Lógica. Lo lógico es

hacer esto en un sub-proceso porque Superglús tratará de buscar una similitud entre la nueva Sentencia Lógica y el resto de la frase.

Entonces comencemos una nueva tabla de Proceso (que debe ser la 5) e insertemos las siguientes entradas:

```
_           _       PARSE                ;Convierte una cadena
                                           a SL
                    MESSAGE 16           ;No es una frase
                                           válida, así que...
                    DONE                 ;El perro no te entiende

SIENTATE    _       ZERO    114          ;¿El perro no está amarrado?
                    SET     114          ;Ahora está sentado quieto
                    MESSAGE 24           ;Díselo al jugador (si está en el
                                           mismo lugar que el perro)
                    DONE

VEN         _       EQ      114 255      ;El perro debe estar sentado
                    CLEAR   114          ;Ahora está normal
                    MESSAGE 18           ;El perro te sigue
                    DONE

_           AQUI    EQ      114 255
                    CLEAR   114
                    MESSAGE 18
                    DONE

_           _       MESSAGE 16           ;Cualquier otra cosa
```

La última entrada se usa para obviar el vocabulario tan limitado que entiende el perro, haciéndole que para cualquier otra frase mueva la cola, por eso se pone la última con _ _.

Recordad que Superglús va buscando en toda la tabla de Procesos, y si no encuentra una igual a la que busca caerá en esta última entrada de "_ _".

PARSE: PARSE le permite a SUPERGLÚS continuar buscando condactos si no se encuentra una Sentencia válida. Hay que tener cuidado aquí, puesto que la actual Sentencia Lógica puede estar un poco enrevesada (el parser intentará sacar algún significado de todos modos) así que normalmente se debe insertar algún mensaje como "Parece no entender" o algo similar, y después un DONE para volver a la acción anterior.

Si se forma una Sentencia válida SUPERGLÚS empezará a buscar las siguientes entradas por una similar, como hacía en la tabla de Respuestas. PARSE debe ser solamente usada en un subproceso que sea llamado desde la tabla de Respuestas, puesto que no tiene ningún significado en otra tabla. Date cuenta de como las entradas VEN y AQUI se hacen cargo de una cantidad de frases que el jugador puede usar después de que el perro esté sentado, porque ambas tienen, una delante y otra detrás, la palabra comodín, o sea, la raya.

Y por último, la entrada "_ _" coge cualquier sentencia que se haya metido en la cadena y para la cual el perro no tenga ninguna respuesta específica.

Selecciona la tabla de Respuestas, para insertar algunas entradas extras que controlen el habla y el dejar caer objetos del árbol. La primera de todas es la entrada que hace que los objetos que se dejen en el árbol caigan al suelo. Esto debe ir entre la entrada que se hace cargo de poner objetos dentro de la bolsa y la entrada normal de "DEJAR _".

Entonces [I PONER _ 1] nos situará en esa posición.

La entrada es:

```
PONER _ AT        8       ;Que el jugador esté en la rama
        WHATO             ;Ya hablaremos de ello
        LT        51 255  ;¿Es un objeto válido?
        EQ        54 254  ;¿Es un objeto que llevas encima?
        MESSAGE   15      ;Avisa que está al pie del árbol
        PUTO      7       ;La pone ahí
        DONE
```

Esto es un ejemplo de cómo se crea una acción automática, viene a ser lo mismo que un AUTOG, etc.

WHATO: Es una acción que mira el primer nombre de la Sentencia Lógica actual en la tabla Objeto-Palabra, y la convierte en el número de un objeto. Este número se pone entonces en la bandera 51. La bandera 51 siempre contiene el número del último objeto que se ha usado, o el último del que Superglús tiene una referencia, y siempre que no esté a cero, las banderas asociadas (números 54 y 57) también se activan.

La bandera 54 lleva la localización actual del objeto. Es conveniente que mires el manual técnico para las banderas 51, 54 y 57.

PUTO: Es una acción que cambia de localidad al objeto últimamente tratado, hacia la localidad que se especifique. O sea, es una acción que cambia el objeto de que se trate hacia la nueva localidad especificada. El Mensaje 15 contiene una raya. Siempre que Superglús encuentra una raya en el texto (sea mensaje o sea localidad) la reemplaza con el objeto actual. Es decir, el mensaje se cambia para tener en cuenta el último objeto que se esté usando.

Ahora una entrada relativamente simple para que se haga cargo de: PONER LA CORREA AL PERRO.

```
PONER CORREA PREP      ENCIMA     ;Para estar seguros de que el jugador teclea eso
             NOUN2     PERRO
             CARRIED   5          ;Que el jugador tenga la correa
             SAME      113    38  ;Que esté en la misma localidad que el perro
             LET       114    1   ;Ahora el perro tiene la correa
             DESTROY   5          ;por lo tanto, el jugador no la puede tener
             MESSAGE   21         ;Se le dice
             DONE
```

En las entradas que siguen, introduciremos un nuevo concepto, Se trata de la modificación de la Sentencia Lógica actual. Nosotros queremos que el juego entienda ambos, ATAR EL PERRO AL BANCO y ATAR LA CORREA AL BANCO puesto que son la misma cosa. Pero tenemos que correa y perro tienen diferente valor como palabras, así que la entrada ATAR PERRO irá primero en la tabla (puesto que su valor es más bajo que la correa). Por lo tanto,

hay que convertir el Nombre en CORREA (55) y permitir que SUPERGLÚS lleve a cabo una entrada que equivalga a ATAR CORREA: Un sistema similar hay que utilizar para DESATAR.

Esto es bastante importante, y lo utilizarás en tus juegos con bastante frecuencia. Hay que poner las entradas:

```
ATAR      PERRO   LET       34      55 ;La bandera 34 es el Nombre de la SL

ATAR      CORREA  PREP      A
                  NOUN2     BANCO
                  AT        4           ;Donde está el banco
                  SAME      113     38  ;El perro está aquí
                  EQ        114     1   ;Tiene la correa puesta
                  PLUS      114     1   ;Ahora atado al banco
                  MESSAGE   22          ;Se informa al jugador sobre ello
                  DONE

ATAR      _       NOTDONE               ;Para asegurarse de que no pueda

DESATAR   PERRO   LET       34      55 ;La bandera 34 es el nombre de la SL

DESATAR   CORREA  AT        4           ;Donde está el banco
                  EQ        114     2   ;Si ya está el perro atado
                  CLEAR     114         ;La bandera indica que está libre
                  MESSAGE   25          ;Díselo al jugador
                  CREATE    5           ;Hay que volver a crear la correa
                  GET       5           ;Dársela al jugador, cogerla

DESATAR   _       NOTDONE               ;Es para asegurarse de que no pueda desatar
                                        otra
```

CREATE: Es una acción que debe ser seguida por un número de objeto. Ello causa que el objeto aparezca en la posición donde está el jugador.

GET: Es una acción que si está seguida por un número de objeto, intenta coger el objeto especificado.

Usamos estas acciones en vez de poner el objeto simplemente en 254, porque puede haber problemas debido al peso o al demasiado número de objetos que se lleven, y entonces estos problemas deben de producir información. Finalmente pondremos las entradas que permitan hablar al perro. Hemos también incluido algunas entradas necesarias para que se te permita hablar al pájaro, pero que él te ignore.

```
DECIR PERRO   SAME      113 38 ;Que esté aquí
              PROCESS   5      ;Se le manda a esa tabla para que haga el trabajo
              DONE

DECIR PAJARO  SAME      112 38
              MESSAGE   8
              DONE
```

```
DECIR _     MESSAGE 23     ;Que pregunte: ¿Quién?
            DONE
```

No ponemos nada para la preposición A, esto permite al jugador recortar su orden si así lo quiere. En líneas generales no se debe hacer una comprobación para una Sentencia Lógica extendida, a menos que sea necesario para diferenciar entre dos frases similares.

Como prueba final, ahora debes jugar la aventura intentando atar al perro, hablarle y todas esas chorradas. Por ejemplo, cuando estemos en el camino cerca del banco del parque, con la correa y el perro, intenta PONER LA CORREA AL PERRO y ATARLO AL BANCO. Luego para desatarlo digamos: DESATA PERRO. Si estamos arriba del árbol con la bolsa, digamos PON TODO EN LA BOLSA y SUELTALA. BAJA y MIRA DENTRO DE LA BOLSA.

Para hacer que el perro se siente teclea: DECIR AL PERRO "SIENTATE".

Para hacerle volver: DECIR AL PERRO "VEN AQUI". ¿Funciona?... Malegro.

10. Mejoras por tí mismo

Hay unos cuantos puntos que convendría arreglar en el juego de demostración para que sirvan como práctica en el uso del sistema.

1. EXAMINA debe responder a todos los objetos aunque sea con una respuesta general como "No veo nada especial en el _"
2. El pájaro debería de verdad volar, escapándose de tí si intentas coger el emparedado mientras está presente. Por ejemplo, se supone que estaría picoteando el emparedado y cualquier pajarito se las piraría.
3. "DESATA _" y "ATA _" deben tener un mensaje más o menos parecido a "¿Atar qué?" y "¿A qué?", porque la forma en que lo pusimos con NOTDONE era una salida bastante fácil.
4. Si el jugador intenta teclear PONER un objeto DENTRO DE LA BOLSA y la bolsa no está presente, tal como lo tenemos de momento lo único que pasará es que dejará caer el objeto, ¿por qué?. Arréglalo.
5. Nada se ha hecho todavía con la antorcha. Las siguientes entradas te permiten encenderla y apagarla (pero debes tener las palabras encender y apagar en el vocabulario):

```
ENCIENDE ANTORCHA CARRIED    1
                  OZERO      1 0
                  OSET       1 0
                  OK
APAGA    ANTORCHA CARRIED    1
                  ONOTZERO   1 0
                  OCLEAR     1 0
                  OK
```

6. Debes mirar todos los condactos que no hemos puesto aquí en la guía técnica, y leer con atención el capítulo sobre claridad y oscuridad. A lo mejor se podría añadir a esta demostración un sótano debajo del pabellón de música. El movimiento debería ser

puesto también en la tabla de Respuestas con una entrada del tipo de: (suponiendo que 9 sea la nueva localidad)

```
BAJAR _ AT    5 ;¿Está el jugador en el pabellón
        SET   0 ;Si la bandera 0=255 entonces es de noche
        GOTO  9 ;Nueva localidad
        DESC
```

7. No debes olvidarte de poner una entrada para SUBIR que limpie la bandera otra vez a 0

8. ¿Qué pasa si el jugador intenta subir al árbol? y ¿cómo sería otra manera de poner esto?. Como clave te diremos que solamente hay una cosa que se puede subir en esa localidad

11. Gráficos

[Este punto está pendiente de hacer en este manual].

12. Notas finales

Este manual ha sido realizado partiendo de una trascripción del manual de SUPERGLÚS para Spectrum realizada por Sien en formato HTML, y ha sido adaptada para Superglús. Esto quiere decir que es posible que haya errores y que algo no funcione del todo bien. Presentamos nuestras disculpas por cualquier error que pudiera aparecer.