DAAD CondActs Reference Manual

Tabla de Contenidos

Condiciones (Conditions)

Acciones (Actions)

Introducción

DAAD (Diseñador de Aventuras de Aventuras AD) es un sistema de programación diseñado específicamente para crear aventuras de texto. Este documento contiene una referencia completa de todos los CondActs (Conditions and Actions) disponibles en DAAD, basado en el manual técnico original de la versión 2.1.

Conceptos Básicos

Parámetros comunes:

- `locno.` = número de localización válido
- `locno+` = número de localización válido o: 252 ("not-created"), 253 ("WORN"), 254 ("CARRIED"), 255 ("HERE")
- `mesno.` = número de mensaje válido
- `sysno.` = número de mensaje del sistema
- `flagno.` = cualquier flag (0-255)
- `objno.` = número de objeto válido

- `value` = valor de 0 a 255
- `word` = palabra del vocabulario o "_" para palabra nula

Indirección: Muchos CondActs permiten indirección en su primer parámetro usando corchetes `[]`. Por ejemplo, `MESSAGE [100]` imprimirá el mensaje cuyo número está almacenado en la Flag 100.

Condiciones de Localización

AT locno. (ID: 0)

Descripción: Verifica si el jugador está en la localización especificada. **Uso:** Control de presencia del jugador en ubicaciones específicas. **Ejemplo:**

```
AT 5    ; ¿El jugador está en la localización 5?
```

NOTAT locno. (ID: 1)

Descripción: Verifica si el jugador NO está en la localización especificada. **Uso:** Control negativo de presencia del jugador. **Ejemplo:**

```
NOTAT 0    ; ¿El jugador NO está en la localización 0?
```

ATGT locno.

Descripción: Verifica si la localización actual es mayor que la especificada. **Uso:** Control de rangos de localizaciones. **Ejemplo:**

```
ATGT 10    ; ¿La localización actual es mayor que 10?
```

ATLT locno.

Descripción: Verifica si la localización actual es menor que la especificada. **Uso:** Control de rangos de localizaciones. **Ejemplo:**

```
ATLT 20    ; ¿La localización actual es menor que 20?
```

Condiciones de Objetos

PRESENT objno.

Descripción: Verifica si el objeto está presente (llevado, vestido o en la localización actual). **Uso:** Control general de disponibilidad de objetos. **Ejemplo:**

```
PRESENT 1    ; ¿El objeto 1 está presente?
MESSAGE 10   ; "Veo la espada aquí"
```

ABSENT objno.

Descripción: Verifica si el objeto NO está presente. **Uso:** Control negativo de disponibilidad de objetos. **Ejemplo:**

```
ABSENT 1     ; ¿El objeto 1 NO está presente?
MESSAGE 11   ; "No veo la espada por aquí"
```

WORN objno.

Descripción: Verifica si el objeto está siendo vestido. **Uso:** Control de estado de vestimenta. **Ejemplo:**

```
WORN 5       ; ¿El objeto 5 está siendo vestido?
MESSAGE 12   ; "Llevo puesto el casco"
```

NOTWORN objno.

Descripción: Verifica si el objeto NO está siendo vestido. **Uso:** Control negativo de estado de vestimenta. **Ejemplo:**

```
NOTWORN 5    ; ¿El objeto 5 NO está siendo vestido?
MESSAGE 13   ; "No llevo puesto el casco"
```

CARRIED objno.

Descripción: Verifica si el objeto está siendo llevado. **Uso:** Control de inventario del jugador. **Ejemplo:**

```
CARRIED 2    ; ¿El objeto 2 está siendo llevado?
MESSAGE 14   ; "Tengo la llave en mis manos"
```

NOTCARR objno.

Descripción: Verifica si el objeto NO está siendo llevado. **Uso:** Control negativo de inventario. **Ejemplo:**

```
NOTCARR 2    ; ¿El objeto 2 NO está siendo llevado?
MESSAGE 15   ; "No tengo la llave"
```

ISAT objno. locno+

Descripción: Verifica si el objeto está en la localización especificada. **Uso:** Control de posición específica de objetos. **Ejemplo:**

```
ISAT 3 10    ; ¿El objeto 3 está en la localización 10?
MESSAGE 16   ; "El tesoro está en la cueva"
```

ISNOTAT objno. locno+

Descripción: Verifica si el objeto NO está en la localización especificada. **Uso:** Control negativo de posición de objetos. **Ejemplo:**

```
ISNOTAT 3 10 ; ¿El objeto 3 NO está en la localización 10?
MESSAGE 17   ; "El tesoro no está en la cueva"
```

Condiciones de Flags

ZERO flagno.

Descripción: Verifica si la flag está en cero. **Uso:** Control de estado inicial o reseteo. **Ejemplo:**

```
ZERO 30      ; ¿La flag 30 está en cero?
MESSAGE 18   ; "La puntuación es cero"
```

NOTZERO flagno.

Descripción: Verifica si la flag NO está en cero. **Uso:** Control de estado activo. **Ejemplo:**

```
NOTZERO 30   ; ¿La flag 30 NO está en cero?
PRINT 30     ; Mostrar la puntuación
```

EQ flagno. value

Descripción: Verifica si la flag es igual al valor especificado. **Uso:** Control de valor específico. **Ejemplo:**

```

```

```
EQ 30 100    ; ¿La flag 30 es igual a 100?
MESSAGE 19   ; "¡Puntuación perfecta!"
```

NOTEQ flagno. value

Descripción: Verifica si la flag NO es igual al valor especificado. **Uso:** Control de valor diferente. **Ejemplo:**

```
NOTEQ 30 0   ; ¿La flag 30 NO es igual a 0?
MESSAGE 20   ; "Tienes algunos puntos"
```

GT flagno. value

Descripción: Verifica si la flag es mayor que el valor especificado. **Uso:** Control de umbral superior. **Ejemplo:**

```
GT 30 50     ; ¿La flag 30 es mayor que 50?
MESSAGE 21   ; "¡Buen progreso!"
```

LT flagno. value

Descripción: Verifica si la flag es menor que el valor especificado. **Uso:** Control de umbral inferior. **Ejemplo:**

```
LT 1 4       ; ¿Llevas menos de 4 objetos?
MESSAGE 22   ; "Puedes llevar más cosas"
```

SAME flagno1 flagno2

Descripción: Verifica si dos flags tienen el mismo valor. **Uso:** Comparación entre flags. **Ejemplo:**

```
SAME 30 31   ; ¿Las flags 30 y 31 tienen el mismo valor?
MESSAGE 23   ; "Los valores coinciden"
```

NOTSAME flagno1 flagno2

Descripción: Verifica si dos flags NO tienen el mismo valor. **Uso:** Comparación negativa entre flags. **Ejemplo:**

```
NOTSAME 30 31 ; ¿Las flags 30 y 31 NO tienen el mismo valor?
MESSAGE 24    ; "Los valores son diferentes"
```

BIGGER flagno1 flagno2

Descripción: Verifica si la primera flag es mayor que la segunda. **Uso:** Comparación de magnitud entre flags. **Ejemplo:**

```
BIGGER 30 31  ; ¿La flag 30 es mayor que la flag 31?
MESSAGE 25    ; "El primer valor es mayor"
```

SMALLER flagno1 flagno2

Descripción: Verifica si la primera flag es menor que la segunda. **Uso:** Comparación de magnitud inversa entre flags. **Ejemplo:**

```
SMALLER 30 31 ; ¿La flag 30 es menor que la flag 31?
MESSAGE 26    ; "El primer valor es menor"
```

Condiciones de Sentencia Lógica

ADJECT1 word

Descripción: Verifica si el primer adjetivo de la sentencia lógica es la palabra especificada. **Uso:** Control de modificadores específicos. **Ejemplo:**

```
ADJECT1 ROJA  ; ¿El primer adjetivo es "ROJA"?
MESSAGE 27    ; "Te refieres a la pelota roja"
```

ADVERB word

Descripción: Verifica si el adverbio de la sentencia lógica es la palabra especificada. **Uso:** Control de modificadores de acción. **Ejemplo:**

```
ADVERB RAPIDAMENTE ; ¿El adverbio es "RAPIDAMENTE"?
MESSAGE 28         ; "Lo haces con prisa"
```

PREP word

Descripción: Verifica si la preposición de la sentencia lógica es la palabra especificada. **Uso:** Control de relaciones espaciales o de acción. **Ejemplo:**

```
PREP CON     ; ¿La preposición es "CON"?
MESSAGE 29   ; "Necesitas especificar con qué"
```

NOUN2 word

Descripción: Verifica si el segundo sustantivo de la sentencia lógica es la palabra especificada. **Uso:** Control de objeto indirecto. **Ejemplo:**

```
NOUN2 CAJA   ; ¿El segundo sustantivo es "CAJA"?
MESSAGE 30   ; "Lo pones en la caja"
```

ADJECT2 word

Descripción: Verifica si el segundo adjetivo de la sentencia lógica es la palabra especificada. **Uso:** Control de modificadores del objeto indirecto. **Ejemplo:**

```
ADJECT2 GRANDE ; ¿El segundo adjetivo es "GRANDE"?
MESSAGE 31     ; "Te refieres a la caja grande"
```

Condiciones de Estado

CHANCE percent

Descripción: Condición aleatoria que se cumple con el porcentaje especificado. **Uso:** Eventos aleatorios y variabilidad en el juego. **Ejemplo:**

```
CHANCE 30    ; 30% de probabilidad
MESSAGE 32   ; "¡Un rayo ilumina el cielo!"
```

ISDONE

Descripción: Verifica si la última tabla procesada terminó ejecutando al menos una acción. **Uso:** Control de éxito/fallo de subprocesos. **Ejemplo:**

```
PROCESS 5    ; Llamar subproceso
ISDONE       ; ¿El subproceso tuvo éxito?
MESSAGE 33   ; "La acción se completó"
```

ISNDONE

Descripción: Verifica si la última tabla procesada terminó sin hacer nada o con NOTDONE. **Uso:** Control de fallo de subprocesos. **Ejemplo:**

```

```

```
PROCESS 5    ; Llamar subproceso
ISNDONE      ; ¿El subproceso falló?
MESSAGE 34   ; "La acción no se pudo completar"
```

HASAT value

Descripción: Verifica si el atributo especificado está activado. **Uso:** Control de atributos de objetos y sistema.
Ejemplo:

```
HASAT WEARABLE ; ¿El objeto actual es vestible?
MESSAGE 35     ; "Puedes vestirte con esto"
```

HASNAT value

Descripción: Verifica si el atributo especificado NO está activado. **Uso:** Control negativo de atributos.
Ejemplo:

```
HASNAT CONTAINER ; ¿El objeto actual NO es contenedor?
MESSAGE 36       ; "No puedes meter cosas ahí"
```

Condiciones de Interacción

INKEY

Descripción: Verifica si el jugador está presionando una tecla. **Uso:** Control de entrada directa del teclado.
Ejemplo:

```
INKEY        ; ¿Se está presionando una tecla?
MESSAGE 37   ; "Tecla presionada"
```

QUIT

Descripción: Pregunta al jugador si está seguro de querer salir. **Uso:** Confirmación de salida del juego.
Ejemplo:

```
QUIT         ; ¿Está seguro de salir?
END          ; Salir del juego
```

Acciones de Manipulación de Objetos

GET objno.

Descripción: Intenta tomar el objeto especificado. **Uso:** Recoger objetos del entorno. **Comportamiento:** Verifica peso, límites de objetos, disponibilidad. **Ejemplo:**

```
GET 1        ; Tomar objeto 1
DONE         ; Terminar entrada
```

DROP objno.

Descripción: Intenta soltar el objeto especificado. **Uso:** Dejar objetos en la localización actual. **Comportamiento:** Verifica si se tiene el objeto, si está vestido. **Ejemplo:**

```
DROP 1       ; Soltar objeto 1
DONE         ; Terminar entrada
```

WEAR objno.

Descripción: Intenta vestir el objeto especificado. **Uso:** Ponerse objetos vestibles. **Comportamiento:** Verifica si es vestible, si se lleva. **Ejemplo:**

```
WEAR 5       ; Vestir objeto 5 (casco)
DONE         ; Terminar entrada
```

REMOVE objno.

Descripción: Intenta quitar el objeto especificado. **Uso:** Quitarse objetos vestidos. **Comportamiento:** Verifica si está vestido, límites de objetos. **Ejemplo:**

```
REMOVE 5     ; Quitar objeto 5
DONE         ; Terminar entrada
```

CREATE objno.

Descripción: Crea el objeto en la localización actual. **Uso:** Hacer aparecer objetos dinámicamente. **Comportamiento:** Coloca el objeto en la localización actual. **Ejemplo:**

```
CREATE 10    ; Crear objeto 10 aquí
MESSAGE 38   ; "¡Aparece un objeto mágico!"
```

DESTROY objno.

Descripción: Destruye el objeto (lo hace desaparecer). **Uso:** Eliminar objetos del juego. **Comportamiento:** Marca el objeto como no creado. **Ejemplo:**

```
DESTROY 10   ; Destruir objeto 10
MESSAGE 39   ; "El objeto se desvanece"
```

SWAP objno1 objno2

Descripción: Intercambia las posiciones de dos objetos. **Uso:** Intercambio de posiciones de objetos. **Comportamiento:** Los objetos intercambian ubicaciones. **Ejemplo:**

```
SWAP 1 2     ; Intercambiar posiciones de objetos 1 y 2
MESSAGE 40   ; "Los objetos cambian de lugar"
```

PLACE objno. locno+

Descripción: Coloca el objeto en la localización especificada. **Uso:** Mover objetos a ubicaciones específicas. **Comportamiento:** Ajusta contadores de objetos llevados. **Ejemplo:**

```
PLACE 1 5    ; Colocar objeto 1 en localización 5
MESSAGE 41   ; "El objeto aparece en la cueva"
```

PUTO locno+

Descripción: Coloca el objeto actual en la localización especificada. **Uso:** Mover el objeto referenciado actualmente. **Comportamiento:** Usa el objeto en flag 51. **Ejemplo:**

```
PUTO 254     ; Colocar objeto actual en inventario (CARRIED)
MESSAGE 42   ; "Ahora lo llevas"
```

PUTIN objno. locno.

Descripción: Intenta meter el objeto en el contenedor especificado. **Uso:** Almacenar objetos en contenedores. **Comportamiento:** Verifica si se tiene el objeto, si está vestido. **Ejemplo:**

```
PUTIN 1 10   ; Meter objeto 1 en contenedor 10
MESSAGE 43   ; "Lo metes en la caja"
```

TAKEOUT objno. locno.

Descripción: Intenta sacar el objeto del contenedor especificado. **Uso:** Extraer objetos de contenedores. **Comportamiento:** Verifica límites de peso y cantidad. **Ejemplo:**

```
TAKEOUT 1 10 ; Sacar objeto 1 del contenedor 10
MESSAGE 44   ; "Lo sacas de la caja"
```

DROPALL

Descripción: Suelta todos los objetos llevados y vestidos. **Uso:** Soltar todo el inventario de una vez. **Comportamiento:** Coloca todos los objetos en la localización actual. **Ejemplo:**

```
DROPALL      ; Soltar todo
MESSAGE 45   ; "Sueltas todo lo que llevas"
```

COPYOO objno1 objno2

Descripción: Copia la posición del primer objeto al segundo. **Uso:** Sincronizar posiciones de objetos. **Comportamiento:** El segundo objeto va donde está el primero. **Ejemplo:**

```
COPYOO 1 2   ; Copiar posición del objeto 1 al 2
MESSAGE 46   ; "Los objetos se reúnen"
```

RESET

Descripción: Coloca todos los objetos en sus posiciones iniciales. **Uso:** Reiniciar estado de objetos. **Comportamiento:** Restaura el estado inicial de todos los objetos. **Ejemplo:**

```
RESET        ; Reiniciar posiciones de objetos
MESSAGE 47   ; "Todo vuelve a su lugar original"
```

Acciones Automáticas de Objetos

AUTOG

Descripción: Versión automática de GET. Busca el objeto por nombre. **Uso:** Tomar objetos por nombre del vocabulario. **Comportamiento:** Busca el objeto por prioridad: aquí, llevado, vestido. **Ejemplo:**

```
COGER ESPADA
AUTOG        ; Tomar la espada automáticamente
DONE
```

AUTOD

Descripción: Versión automática de DROP. Busca el objeto por nombre. **Uso:** Soltar objetos por nombre del vocabulario. **Comportamiento:** Busca el objeto por prioridad: llevado, vestido, aquí. **Ejemplo:**

```
SOLTAR ESPADA
AUTOD        ; Soltar la espada automáticamente
DONE
```

AUTOW

Descripción: Versión automática de WEAR. Busca el objeto por nombre. **Uso:** Vestir objetos por nombre del vocabulario. **Comportamiento:** Busca el objeto por prioridad: llevado, vestido, aquí. **Ejemplo:**

```
VESTIR CASCO
AUTOW        ; Vestir el casco automáticamente
DONE
```

AUTOR

Descripción: Versión automática de REMOVE. Busca el objeto por nombre. **Uso:** Quitar objetos por nombre del vocabulario. **Comportamiento:** Busca el objeto por prioridad: vestido, llevado, aquí. **Ejemplo:**

```
QUITAR CASCO
AUTOR        ; Quitar el casco automáticamente
DONE
```

AUTOP locno.

Descripción: Versión automática de PUTIN. Busca el objeto por nombre. **Uso:** Meter objetos en contenedores por nombre. **Comportamiento:** Busca el objeto por prioridad: llevado, vestido, aquí. **Ejemplo:**

```
METER LLAVE CAJA
AUTOP 10     ; Meter la llave en la caja (objeto 10)
DONE
```

AUTOT locno.

Descripción: Versión automática de TAKEOUT. Busca el objeto por nombre. **Uso:** Sacar objetos de contenedores por nombre. **Comportamiento:** Busca en el contenedor primero. **Ejemplo:**

```
SACAR LLAVE CAJA
AUTOT 10     ; Sacar la llave de la caja (objeto 10)
DONE
```

COPYOF objno. flagno.

Descripción: Copia la posición del objeto a una flag. **Uso:** Guardar posición de objeto para procesamiento. **Comportamiento:** La flag recibe la localización del objeto. **Ejemplo:**

```
COPYOF 1 100 ; Copiar posición del objeto 1 a flag 100
PRINT 100    ; Mostrar la posición
```

COPYFO flagno. objno.

Descripción: Copia el valor de una flag a la posición del objeto. **Uso:** Mover objeto a posición calculada. **Comportamiento:** El objeto va a la localización indicada en la flag. **Ejemplo:**

```
COPYFO 100 1 ; Mover objeto 1 a posición almacenada en flag 100
MESSAGE 48   ; "El objeto se mueve"
```

WHATO

Descripción: Busca el objeto por nombre y lo hace el objeto actual. **Uso:** Identificar objeto por nombre para acciones posteriores. **Comportamiento:** Establece el objeto en las flags 51, 54-57. **Ejemplo:**

```
WHATO        ; Identificar objeto por nombre
PRINT 51     ; Mostrar número del objeto identificado
```

SETCO objno.

Descripción: Establece el objeto actual. **Uso:** Cambiar el objeto de referencia. **Comportamiento:** Establece el objeto en las flags correspondientes. **Ejemplo:**

```
SETCO 5      ; Establecer objeto 5 como actual
MESSAGE 49   ; "Ahora trabajas con el casco"
```

WEIGH objno. flagno.

Descripción: Calcula el peso total del objeto y lo guarda en una flag. **Uso:** Determinar peso de objetos y contenedores. **Comportamiento:** Incluye peso de contenidos si es contenedor. **Ejemplo:**

```
WEIGH 10 100 ; Calcular peso del objeto 10 y guardarlo en flag 100
PRINT 100    ; Mostrar el peso
```

Acciones de Flags

SET flagno.

Descripción: Establece la flag al valor máximo (255). **Uso:** Activar flags booleanas. **Comportamiento:** Flag = 255. **Ejemplo:**

```
SET 100      ; Activar flag 100
MESSAGE 50   ; "Función activada"
```

CLEAR flagno.

Descripción: Establece la flag a cero. **Uso:** Desactivar flags o reiniciar valores. **Comportamiento:** Flag = 0. **Ejemplo:**

```
CLEAR 100    ; Desactivar flag 100
MESSAGE 51   ; "Función desactivada"
```

LET flagno. value

Descripción: Establece la flag al valor especificado. **Uso:** Asignar valores específicos a flags. **Comportamiento:** Flag = value. **Ejemplo:**

```
LET 30 100   ; Establecer puntuación a 100
MESSAGE 52   ; "¡Puntuación máxima!"
```

PLUS flagno. value

Descripción: Suma el valor a la flag. **Uso:** Incrementar valores de flags. **Comportamiento:** Flag = Flag + value (máximo 255). **Ejemplo:**

```
PLUS 30 10   ; Sumar 10 puntos
MESSAGE 53   ; "¡Ganas 10 puntos!"
```

MINUS flagno. value

Descripción: Resta el valor de la flag. **Uso:** Decrementar valores de flags. **Comportamiento:** Flag = Flag - value (mínimo 0). **Ejemplo:**

```
MINUS 30 5   ; Restar 5 puntos
MESSAGE 54   ; "Pierdes 5 puntos"
```

ADD flagno1 flagno2

Descripción: Suma el valor de la primera flag a la segunda. **Uso:** Operaciones aritméticas entre flags. **Comportamiento:** Flag2 = Flag2 + Flag1 (máximo 255). **Ejemplo:**

```
ADD 31 30    ; Sumar flag 31 a flag 30
MESSAGE 55   ; "Valores combinados"
```

SUB flagno1 flagno2

Descripción: Resta el valor de la primera flag de la segunda. **Uso:** Operaciones aritméticas entre flags. **Comportamiento:** Flag2 = Flag2 - Flag1 (mínimo 0). **Ejemplo:**

```
SUB 31 30    ; Restar flag 31 de flag 30
MESSAGE 56   ; "Valores restados"
```

COPYFF flagno1 flagno2

Descripción: Copia el valor de la primera flag a la segunda. **Uso:** Duplicar valores de flags. **Comportamiento:** Flag2 = Flag1. **Ejemplo:**

```
COPYFF 30 31 ; Copiar puntuación a respaldo
MESSAGE 57   ; "Puntuación guardada"
```

COPYBF flagno1 flagno2

Descripción: Copia el valor de la segunda flag a la primera. **Uso:** Copia inversa para usar con indirección. **Comportamiento:** Flag1 = Flag2. **Ejemplo:**

```
COPYBF 30 31 ; Copiar de flag 31 a flag 30
MESSAGE 58   ; "Valor restaurado"
```

RANDOM flagno.

Descripción: Genera un número aleatorio (1-100) en la flag. **Uso:** Generar valores aleatorios para decisiones. **Comportamiento:** Flag = random(1, 100). **Ejemplo:**

```
RANDOM 100   ; Generar número aleatorio
GT 100 50    ; ¿Mayor que 50?
MESSAGE 59   ; "¡Tienes suerte!"
```

MOVE flagno.

Descripción: Usa el verbo actual para mover desde la localización en la flag. **Uso:** Movimiento de PSIs o jugador programático. **Comportamiento:** Consulta tabla de conexiones. **Ejemplo:**

```
MOVE 38      ; Intentar mover jugador
MESSAGE 60   ; "Te mueves"
```

Acciones de Movimiento

GOTO locno.

Descripción: Mueve el jugador a la localización especificada. **Uso:** Teletransporte o movimiento directo. **Comportamiento:** Establece flag 38 al valor de localización. **Ejemplo:**

```
GOTO 10      ; Ir a localización 10
DESC 10      ; Describir nueva localización
```

WEIGHT flagno.

Descripción: Calcula el peso total de objetos llevados y vestidos. **Uso:** Control de límites de peso. **Comportamiento:** Incluye peso de contenidos de contenedores. **Ejemplo:**

```
WEIGHT 100   ; Calcular peso llevado
GT 100 50    ; ¿Peso mayor que 50?
MESSAGE 61   ; "Llevas demasiado peso"
```

ABILITY value1 value2

Descripción: Establece límites de objetos y peso del jugador. **Uso:** Configurar capacidades del jugador. **Comportamiento:** Flag 37 = value1, Flag 52 = value2. **Ejemplo:**

```
ABILITY 6 75 ; Máximo 6 objetos, peso 75
MESSAGE 62   ; "Te sientes más fuerte"
```

Acciones de Pantalla

MODE option

Descripción: Cambia el modo de operación de la ventana actual. **Uso:** Control de visualización de texto. **Comportamiento:** Opciones: 1=conjunto superior, 2=sin "More...". **Ejemplo:**

```
MODE 3       ; Conjunto superior + sin "More..."
MESSAGE 63   ; "Modo especial activado"
```

INPUT stream option

Descripción: Establece el flujo de entrada y opciones. **Uso:** Control de entrada de texto. **Comportamiento:** Opciones: 1=limpiar, 2=reimprimir, 4=recordar. **Ejemplo:**

```
INPUT 0 1    ; Entrada del flujo actual, limpiar después
MESSAGE 64   ; "Entrada configurada"
```

TIME duration option

Descripción: Establece timeout para entrada de texto. **Uso:** Control de tiempo límite. **Comportamiento:** Duración en segundos, opciones para diferentes situaciones. **Ejemplo:**

```
TIME 30 7    ; 30 segundos, todas las situaciones
MESSAGE 65   ; "Tiempo límite establecido"
```

WINDOW window

Descripción: Selecciona la ventana activa (0-7). **Uso:** Control de ventanas múltiples. **Comportamiento:** Cambia flujo de salida. **Ejemplo:**

```
WINDOW 1     ; Cambiar a ventana 1
MESSAGE 66   ; "Mensaje en ventana 1"
```

WINAT line col

Descripción: Posiciona la ventana actual en línea y columna. **Uso:** Control de posicionamiento de ventanas. **Comportamiento:** Ajusta posición con recorte. **Ejemplo:**

```

```

```
WINAT 5 10   ; Posicionar ventana en línea 5, columna 10
MESSAGE 67   ; "Ventana reposicionada"
```

WINSIZE height width

Descripción: Establece el tamaño de la ventana actual. **Uso:** Control de dimensiones de ventanas. **Comportamiento:** Ajusta tamaño con recorte. **Ejemplo:**

```
WINSIZE 10 40 ; Ventana de 10 líneas por 40 columnas
MESSAGE 68    ; "Ventana redimensionada"
```

CENTRE

Descripción: Centra la ventana actual para el ancho de pantalla. **Uso:** Alineación automática de ventanas. **Comportamiento:** Solo afecta posición horizontal. **Ejemplo:**

```
CENTRE       ; Centrar ventana
MESSAGE 69   ; "Ventana centrada"
```

CLS

Descripción: Limpia la ventana actual. **Uso:** Limpiar pantalla o ventana específica. **Comportamiento:** Borra contenido de ventana. **Ejemplo:**

```
CLS          ; Limpiar ventana
MESSAGE 70   ; "Pantalla limpia"
```

SAVEAT

Descripción: Guarda la posición actual de impresión. **Uso:** Mantener posición para restaurar después. **Comportamiento:** Guarda posición en pila interna. **Ejemplo:**

```
SAVEAT       ; Guardar posición
WINDOW 2     ; Cambiar a otra ventana
MESSAGE 71   ; "Mensaje temporal"
BACKAT       ; Restaurar posición
```

BACKAT

Descripción: Restaura la posición de impresión guardada. **Uso:** Volver a posición anterior. **Comportamiento:** Restaura posición desde pila. **Ejemplo:**

```
BACKAT       ; Restaurar posición
MESSAGE 72   ; "De vuelta a la posición original"
```

PAPER colour

Descripción: Establece el color de fondo. **Uso:** Control de colores de fondo. **Comportamiento:** Usa tabla de colores específica de máquina. **Ejemplo:**

```
PAPER 4      ; Color de fondo 4
MESSAGE 73   ; "Fondo cambiado"
```

INK colour

Descripción: Establece el color de tinta (texto). **Uso:** Control de colores de texto. **Comportamiento:** Usa tabla de colores específica de máquina. **Ejemplo:**

```
INK 2        ; Color de tinta 2
MESSAGE 74   ; "Texto en color"
```

BORDER colour

Descripción: Establece el color del borde de pantalla. **Uso:** Control de borde de pantalla. **Comportamiento:** Específico de máquina. **Ejemplo:**

```
BORDER 1     ; Borde color 1
MESSAGE 75   ; "Borde cambiado"
```

BEEP value1 value2

Descripción: Produce un sonido o pitido. **Uso:** Efectos sonoros simples. **Comportamiento:** Genera audio usando altavoz del sistema. Los parámetros controlan frecuencia y duración (específico de máquina). **Ejemplo:**

```
BEEP 100 50  ; Pitido con frecuencia 100, duración 50
MESSAGE 76   ; "¡Sonido emitido!"
```

PRINTAT line col

Descripción: Establece la posición de impresión en la ventana. **Uso:** Control de posición de cursor. **Comportamiento:** Si está fuera de ventana, va a esquina superior izquierda. **Ejemplo:**

```
PRINTAT 5 10 ; Posición línea 5, columna 10
MESSAGE 76   ; "Mensaje en posición específica"
```

TAB col

Descripción: Mueve el cursor a la columna especificada. **Uso:** Alineación horizontal de texto. **Comportamiento:** En la línea actual. **Ejemplo:**

```
TAB 20       ; Ir a columna 20
MESSAGE 77   ; "Texto tabulado"
```

SPACE

Descripción: Imprime un espacio. **Uso:** Espaciado manual de texto. **Comportamiento:** Más corto que un mensaje con espacio. **Ejemplo:**

```
MESSAGE 78   ; "Palabra"
SPACE        ; Espacio
MESSAGE 79   ; "Otra palabra"
```

NEWLINE

Descripción: Imprime retorno de carro y avance de línea. **Uso:** Salto de línea manual. **Comportamiento:** Nueva línea. **Ejemplo:**

```
MESSAGE 80   ; "Primera línea"
NEWLINE      ; Nueva línea
MESSAGE 81   ; "Segunda línea"
```

Acciones de Entrada/Salida

MES mesno.

Descripción: Imprime el mensaje especificado. **Uso:** Mostrar texto sin salto de línea. **Comportamiento:** Solo imprime el mensaje. **Ejemplo:**

```
MES 10       ; Imprimir mensaje 10
SPACE        ; Espacio
MES 11       ; Imprimir mensaje 11
```

MESSAGE mesno.

Descripción: Imprime el mensaje especificado y salta de línea. **Uso:** Mostrar texto con salto de línea. **Comportamiento:** Imprime mensaje + NEWLINE. **Ejemplo:**

```
MESSAGE 82   ; "¡Hola mundo!"
MESSAGE 83   ; "Segunda línea"
```

SYSMESS sysno.

Descripción: Imprime el mensaje del sistema especificado. **Uso:** Mostrar mensajes estándar del sistema. **Comportamiento:** Mensajes predefinidos del intérprete. **Ejemplo:**

```
SYSMESS 8    ; "No puedo hacer eso"
DONE
```

DESC locno.

Descripción: Imprime la descripción de la localización sin salto de línea. **Uso:** Mostrar texto de localización. **Comportamiento:** Solo el texto descriptivo. **Ejemplo:**

```
DESC 38      ; Describir localización actual
NEWLINE      ; Saltar línea
LISTOBJ      ; Listar objetos presentes
```

Acciones de Impresión

PRINT flagno.

Descripción: Imprime el valor decimal de la flag. **Uso:** Mostrar valores numéricos. **Comportamiento:** Sin espacios adicionales. **Ejemplo:**

```
MES 84       ; "Puntuación: "
PRINT 30     ; Mostrar valor de flag 30
MESSAGE 85   ; " puntos"
```

DPRINT flagno.

Descripción: Imprime un número de dos bytes (flag y flag+1). **Uso:** Mostrar valores grandes (0-65535). **Comportamiento:** (flag+1) * 256 + flag. **Ejemplo:**

```
MES 86       ; "Valor grande: "
DPRINT 100   ; Mostrar valor de flags 100-101
```

```
NEWLINE
```

Acciones de Listas

LISTOBJ

Descripción: Lista los objetos presentes en la localización actual. **Uso:** Mostrar objetos disponibles. **Comportamiento:** Si hay objetos, imprime SM1 + lista. **Ejemplo:**

```
DESC 38      ; Describir localización
LISTOBJ      ; Listar objetos presentes
```

LISTAT locno+

Descripción: Lista los objetos en la localización especificada. **Uso:** Mostrar contenido de contenedores o localizaciones. **Comportamiento:** Si no hay objetos, imprime SM53 ("nothing"). **Ejemplo:**

```
MES 87       ; "En la caja hay: "
LISTAT 10    ; Listar objetos en contenedor 10
```

Acciones de Guardado

SAVE opt

Descripción: Guarda el estado actual del juego. **Uso:** Sistema de guardado. **Comportamiento:** opt: 0=normal, 1=cinta, 2=disco. **Ejemplo:**

```
SAVE 2       ; Guardar en disco
MESSAGE 88   ; "Juego guardado"
```

LOAD opt

Descripción: Carga un estado guardado del juego. **Uso:** Sistema de carga. **Comportamiento:** opt: 0=normal, 1=cinta, 2=disco. **Ejemplo:**

```
LOAD 2       ; Cargar desde disco
MESSAGE 89   ; "Juego cargado"
```

RAMSAVE

Descripción: Guarda el estado del juego en memoria. **Uso:** Guardado temporal en RAM. **Comportamiento:** Volátil, se pierde al apagar. **Ejemplo:**

```
RAMSAVE      ; Guardar en memoria
MESSAGE 90   ; "Estado guardado en memoria"
```

RAMLOAD flagno.

Descripción: Carga el estado guardado en memoria. **Uso:** Restaurar estado temporal. **Comportamiento:** Parámetro especifica última flag a cargar. **Ejemplo:**

```
RAMLOAD 254  ; Cargar estado de memoria
MESSAGE 91   ; "Estado restaurado"
```

Acciones de Control

ANYKEY

Descripción: Espera que se presione cualquier tecla. **Uso:** Pausar el juego esperando entrada. **Comportamiento:** Imprime SM16 y espera tecla. **Ejemplo:**

```
MESSAGE 92   ; "¡Evento importante!"
ANYKEY       ; Esperar tecla
```

PAUSE value

Descripción: Pausa durante el tiempo especificado. **Uso:** Pausas automáticas. **Comportamiento:** value/50 segundos (0 = 256/50 segundos). **Ejemplo:**

```
PAUSE 100    ; Pausar 2 segundos
MESSAGE 93   ; "Después de la pausa"
```

PARSE n

Descripción: Analiza entrada del jugador. **Uso:** Control del analizador sintáctico. **Comportamiento:** n: 0=línea principal, 1=cadena entrecomillada. **Ejemplo:**

```
PARSE 0      ; Analizar entrada principal
MESSAGE 94   ; "Entrada procesada"
```

NEWTEXT

Descripción: Fuerza la pérdida de frases restantes en línea de entrada. **Uso:** Cancelar entrada múltiple tras errores. **Comportamiento:** Limpia buffer de entrada. **Ejemplo:**

```
NEWTEXT      ; Cancelar entrada restante
MESSAGE 95   ; "Entrada cancelada"
```

SYNONYM verb noun

Descripción: Sustituye verbo y sustantivo en sentencia lógica. **Uso:** Normalizar entrada del jugador. **Comportamiento:** Reemplaza palabras en sentencia actual. **Ejemplo:**

```
SYNONYM ENCENDER ANTORCHA ; Normalizar comando
MESSAGE 96 ; "Enciendes la antorcha"
```

PROCESS procno.

Descripción: Llama a la tabla de proceso especificada. **Uso:** Llamadas a subrutinas. **Comportamiento:** Llamada verdadera con retorno. **Ejemplo:**

```
PROCESS 5    ; Llamar proceso 5
MESSAGE 97   ; "Proceso completado"
```

REDO

Descripción: Reinicia la tabla actual. **Uso:** Bucles y repeticiones. **Comportamiento:** Vuelve al inicio de la tabla. **Ejemplo:**

```
REDO         ; Reiniciar tabla actual
```

DOALL locno+

Descripción: Implementa comandos tipo "ALL". **Uso:** Ejecutar acción en todos los objetos. **Comportamiento:** Itera sobre objetos en localización. **Ejemplo:**

```
COGER TODO
DOALL 255    ; Coger todo en localización actual
AUTOG        ; Coger cada objeto
```

SKIP distance

Descripción: Salta a otra entrada en la tabla. **Uso:** Control de flujo y saltos. **Comportamiento:** distance: -128 a 128, o etiqueta local. **Ejemplo:**

```
SKIP 2       ; Saltar 2 entradas adelante
MESSAGE 98   ; "Esta línea se salta"
MESSAGE 99   ; "Esta también"
MESSAGE 100  ; "Esta se ejecuta"
```

Acciones de Llamadas

EXTERN value

Descripción: Llama a rutina externa con parámetro. **Uso:** Extensiones en ensamblador. **Comportamiento:** Dirección establecida por enlazado. **Ejemplo:**

```
EXTERN 5     ; Llamar rutina externa con parámetro 5
MESSAGE 101  ; "Rutina externa ejecutada"
```

CALL address

Descripción: Ejecuta código en dirección especificada. **Uso:** Llamadas a código máquina. **Comportamiento:** Salta a dirección de memoria. **Ejemplo:**

```
CALL 32768   ; Ejecutar código en dirección 32768
MESSAGE 102  ; "Código ejecutado"
```

SFX value1 value2

Descripción: Acción para efectos de sonido. **Uso:** Reproducir sonidos. **Comportamiento:** Escribe value1 en registro value2 del chip de sonido. **Ejemplo:**

```
SFX 128 1    ; Efecto de sonido
MESSAGE 103  ; "¡Sonido reproducido!"
```

GFX value1 value2

Descripción: Acción para extensiones gráficas. **Uso:** Control gráfico avanzado. **Comportamiento:** En 16-bit controla cambio de pantallas. **Ejemplo:**

```
GFX 1 0      ; Cambiar modo gráfico
MESSAGE 104  ; "Modo gráfico cambiado"
```

Acciones de Gráficos

PICTURE picno

Descripción: Carga la imagen especificada en el buffer. **Uso:** Preparar imágenes para mostrar. **Comportamiento:** Si no existe la imagen, salta a siguiente entrada. **Ejemplo:**

```
PICTURE 5    ; Cargar imagen 5
DISPLAY 0    ; Mostrar imagen
```

DISPLAY value

Descripción: Muestra la imagen buffered o limpia área. **Uso:** Mostrar gráficos en pantalla. **Comportamiento:** value=0 muestra imagen, value!=0 limpia área. **Ejemplo:**

```
DISPLAY 0    ; Mostrar imagen cargada
MESSAGE 105  ; "Imagen mostrada"
```

MOUSE option

Descripción: Manejo básico de ratón (en preparación). **Uso:** Control de ratón (sistema experimental). **Comportamiento:** Esqueleto para sistema futuro. **Ejemplo:**

```
MOUSE 1      ; Activar soporte de ratón
MESSAGE 106  ; "Ratón activado"
```

Acciones de Finalización

RESTART

Descripción: Reinicia completamente el juego. **Uso:** Reiniciar aventura. **Comportamiento:** Cancela DOALL y subprocesos, va a proceso 0. **Ejemplo:**

```
RESTART      ; Reiniciar juego
```

END

Descripción: Termina el juego preguntando si quiere jugar otra vez. **Uso:** Final del juego. **Comportamiento:** Pregunta si quiere jugar de nuevo. **Ejemplo:**

```
MESSAGE 107  ; "¡Has ganado!"
END          ; Terminar juego
```

EXIT value

Descripción: Sale del juego o reinicia. **Uso:** Salida controlada. **Comportamiento:** value=0 sale, value!=0 reinicia. **Ejemplo:**

```
EXIT 0       ; Salir al sistema operativo
```

DONE

Descripción: Termina la tabla actual marcando que se ejecutó una acción. **Uso:** Finalizar entrada exitosamente. **Comportamiento:** Retorna con flag "hecho". **Ejemplo:**

```
GET 1        ; Coger objeto
DONE         ; Terminar entrada exitosamente
```

NOTDONE

Descripción: Termina la tabla actual marcando que NO se ejecutó acción. **Uso:** Finalizar entrada sin éxito. **Comportamiento:** Retorna con flag "no hecho". **Ejemplo:**

```
NOTDONE      ; Terminar sin ejecutar acción
```

OK

Descripción: Imprime "OK" y termina la entrada. **Uso:** Respuesta afirmativa simple. **Comportamiento:** Imprime SM15 + DONE. **Ejemplo:**

```
OK           ; Responder "OK" y terminar
```

Flags del Sistema

Flags Importantes

- **Flag 0**: Indica si el juego está oscuro (usado con objeto 0)
- **Flag 1**: Cantidad de objetos que lleva el jugador
- **Flag 29**: Flags de gráficos y opciones de pantalla
- **Flag 30**: Puntuación (por convención)

- **Flag 33**: Verbo de la sentencia lógica actual
- **Flag 34**: Primer sustantivo de la sentencia lógica
- **Flag 37**: Máximo número de objetos que puede llevar
- **Flag 38**: Localización actual del jugador
- **Flag 51**: Objeto actualmente referenciado
- **Flag 52**: Fuerza del jugador (peso máximo)

Ejemplo de Uso de Flags del Sistema

```
; Comprobar si el jugador puede llevar más objetos
LT 1 37      ; ¿Lleva menos del máximo?
GET 1        ; Coger objeto
DONE

; Si no puede llevar más
MESSAGE 108  ; "No puedes llevar más cosas"
DONE
```

Ejemplos Avanzados

Sistema de Puntuación

```
; Proceso para dar puntos por primera vez
PROCESS 10   ; Proceso de puntuación
ZERO 64      ; ¿Es la primera vez? (flag 64 = marcador)
PLUS 30 50   ; Sumar 50 puntos
SET 64       ; Marcar como hecho
MESSAGE 109  ; "¡Ganas 50 puntos!"
DONE

; No dar puntos si ya se hizo
MESSAGE 110  ; "Ya hiciste esto antes"
DONE
```

Sistema de Inventario Inteligente

```
; Comando INVENTARIO mejorado
INVENTARIO _
ZERO 1       ; ¿No llevas nada?
MESSAGE 111  ; "No llevas nada"
DONE

; Mostrar objetos llevados
MESSAGE 112  ; "Llevas:"
LISTAT 254   ; Listar objetos llevados
DONE
```

Sistema de Contenedores

```
; Comando ABRIR CAJA
ABRIR CAJA
PRESENT 10   ; ¿Está presente la caja?
HASAT CONTAINER ; ¿Es realmente un contenedor?
MESSAGE 113  ; "Abres la caja. Dentro hay:"
LISTAT 10    ; Mostrar contenido
DONE

; Error si no es contenedor
MESSAGE 114  ; "No puedes abrir eso"
DONE
```

Sistema de Movimiento Condicional

```
; Movimiento al norte con condiciones
NORTE _
AT 5         ; ¿Estás en localización 5?
ZERO 65      ; ¿La puerta está cerrada? (flag 65)
MESSAGE 115  ; "La puerta está cerrada"
DONE

; Movimiento exitoso
GOTO 6       ; Ir al norte
DESC 6       ; Describir nueva localización
DONE
```

Notas Importantes

1. **Indirección**: Muchos CondActs permiten indirección en su primer parámetro usando `[]`.
2. **Salida de Tablas**: Los CondActs de salida (DONE, NOTDONE, etc.) terminan el procesamiento de la tabla actual.
3. **Mensajes del Sistema**: Los mensajes del sistema (SM) están predefinidos y pueden variar según el idioma.
4. **Límites**: Los valores de flags están limitados a 0-255.
5. **Objetos Especiales**: El objeto 0 se considera fuente de luz por defecto.

Convenciones de Programación

Estructura Típica de Proceso 0

```
; Proceso principal
_  _  PARSE 0     ; Analizar entrada
      PROCESS 1  ; Procesar comandos
      REDO       ; Repetir bucle

; Si no se procesa nada
_  _  SYSMESS 8  ; "No puedo hacer eso"
      REDO       ; Repetir bucle
```

Manejo de Errores

```
; Verificar condiciones antes de actuar
COGER _
WHATO        ; Identificar objeto
PRESENT 51   ; ¿Está presente?
AUTOG        ; Coger automáticamente
DONE

; Error si no está presente
MESSAGE 116  ; "No veo eso aquí"
DONE
```

Uso de Subprocesos

```
; Llamada a subproceso con verificación
PROCESS 5    ; Llamar subproceso
ISDONE       ; ¿Tuvo éxito?
MESSAGE 117  ; "Acción completada"
DONE

; Manejo de fallo
MESSAGE 118  ; "No se pudo completar"
DONE
```

Este documento proporciona una referencia completa de todos los CondActs disponibles en DAAD. Cada CondAct incluye su descripción, uso típico, comportamiento detallado y ejemplos prácticos de implementación.