# Lenguaje de definición de archivo XML Especificación



El archivo de definición de lenguaje (lang\_def.xml) describe dos partes:

* + bloque de lenguaje
  + Área de trabajo, que es el entorno de programación para el lenguaje de bloque especificado.

(Nota:. Los archivos lang\_def.dtd especifica las reglas, elementos, atributos dentro lang\_def.xml)

# Especificación del idioma Block

## 2.1 Bloque de Género

Un género bloque se describen las propiedades que definen un conjunto común de bloques. Por ejemplo,

*fd es un género de bloques que describe todos los bloques hacia adelante en StarLogo. Para cada bloque en su*

bloquear idioma, debe especificar un BlockGenus.

<BlockGenus Name="fd" kind="command" initlabel="forward" color="255 0 0">

<description>

Agentes <text> mueve <arg n="1"/> pasos hacia adelante. </ text>

<arg-description n="1" name="steps"> Número de pasos para mover. </ Arg-description>

</ Description>

<BlockConnectors>

<BlockConnector Label="steps" connector-kind="socket" connector-type="number">

<DefaultArg Genus-name="number" label="1"> </ DefaultArg>

</ BlockConnector>

</ BlockConnectors>

</ BlockGenus>

**Figura 1: El bloque delantero y su bloque de especificación género fd.**

|  |  |
| --- | --- |
| **Bloque Atributos Género** | **Descripción** |
| nombre | Nombre único para el género |
| tipo | Bloque clase género. En la actualidad, codeblocks utiliza tres tipos: comando, datos, variables |
| initlabel | La etiqueta inicial de un bloque de este género. |
| color | El color RGB para un bloque de este género. |
| editable etiqueta | Si "sí", la etiqueta de bloque es editable por el usuario. Salvo que se especifique en el bloque género, editable-label = "no" por defecto (es decir, género fd no es editable.) |
| etiqueta única | Si "sí", cada instancia de bloque de este género debe tener una etiqueta de bloque único. Se establece en "no" por defecto. |
| es la etiqueta de valor | Si responde "sí", su etiqueta de bloque determina el valor de bloqueo. Este atributo se aplica a los bloques de datos (es decir, número, picadura, true, false). Se establece en "no" por defecto. |
| etiqueta-prefix | Una cadena de preceder siempre su etiqueta de bloque |

|  |  |
| --- | --- |
| etiqueta-suffix | Una cadena de triunfar siempre su etiqueta de bloque |
| sockets expandible | Si "sí", zócalos ampliar cada vez que un bloque está conectado a ellos. Se establece en "no" por defecto. |
| es-starter | Si "sí", este género bloque comienza pilas de bloques de código.  Se establece en "no" por defecto. |
| es terminador | Si "sí", este género bloque termina pilas de bloques de final.  Se establece en "no" por defecto. |

* + 1. *Descripción*

El elemento de descripción especifica el formato y el texto de una descripción fácil de usar de

los bloques dentro de este género.

(Nota: esto no está completamente especificado todavía.)

* + 1. *BlockConnectors*

Los BlockConnectors especifica las propiedades de todos los puntos de conexión de un bloque

género. Hay cuatro tipos de conectores, que difieren en su significado y su ubicación:

* + - * Plug: devuelve un valor. Reside en el lado izquierdo de un bloque.
      * Socket: Lee un valor o ramas del flujo de ejecución. Reside en el lado derecho

de un bloque.

* + - * Antes: Se conecta al comando anterior. Reside en el lado superior de un bloque.
      * Después: Conecta con el siguiente comando. Reside en el lado inferior de un bloque.

Cada conector tiene un tipo que determina su forma y los datos (número, cadena, valor booleano, polimórfica) o el tipo de comando que se lee / Devoluciones / conecta. Conectores de antes y después son siempre de tipo comando. Un conector puede ser de cualquier tipo. Del mismo modo, un plugin devuelve cualquier tipo, a excepción de comandos.

Los enchufes y los conectores se especifican en el archivo xml. Conectores antes y después se generan dinámicamente, dependiendo de si el tipo es del género de comandos o si el género es un motor de arranque o de terminación.

**Figura 2: El género y si su bloque tiene dos zócalos:. ". Entonces" uno booleanos con la etiqueta "test" y un comando de marcado El género y verdadero y su bloque tiene un enchufe booleano refleja en ambos lados**



|  |  |
| --- | --- |
| **Atributos BlockConnector** | **Descripción** |
| etiqueta | Etiqueta que se encuentra en el conector del bloque. Por ejemplo, los pasos es una etiqueta de zócalo delantero de. |
| etiqueta editable | Si "sí", la etiqueta del conector es editable por el usuario. Se establece en "no" por defecto. |
| conector tipo | Determina si el conector es un "enchufe" o un "socket". |

|  |  |
| --- | --- |
|  | Conectores antes y después se generan dinámicamente dependiendo del tipo género, motor de arranque, o propiedades de terminación. |
| conector de tipo | De datos (número, cadena, booleano, polimórfica) o escriba command. |
| Tipo de posición | Determina la posición de este conector. solo: ya sea a la izquierda oa la derecha en función de espejo tipo: conector se refleja en el lado derecho del bloque  fondo: zócalos se colocan en la parte inferior del bloque |

Algunos bloques tienen argumentos por defecto o bloques que están conectados a los conectores cuando el bloque se arrastra sobre el lienzo bloque. Por ejemplo, el bloque delantero tiene un argumento predeterminado, el número de bloque con su valor establecido en 1. Los argumentos por defecto puede ser conveniente, especialmente si el argumento predeterminado es una elección común para conectarse a un bloque.

|  |  |
| --- | --- |
| **Attribues parámetro por defecto default** | **Descripción** |
| género-nombre | el nombre del género de bloques del argumento por defecto |
| etiqueta | la etiqueta (si género es editable) del argumento |

* + 1. *BlockStubs (Avanzado)*

Algunos géneros de bloque pueden haber generado dinámicamente bloques cuando su instancia de bloque es

arrastró sobre el lienzo bloque. Estos bloques generados dinámicamente se llaman Stubs, y hacen referencia al bloque principal que se crearon. Ejemplos de talones son

getters y setters para las variables. El género bloque agente-var-booleano especifica a continuación tiene dos talones: un getter y setter.

<BlockGenus Name="agent-var-boolean" kind="variable" initlabel="agent boolean" editable-label="yes" color="65 105 225">

...

<BlockConnectors>

Connector-kind="socket" <BlockConnector connector-type="boolean">

</ BlockConnector>

</ BlockConnectors>

<Stubs>

<Stub Stub-genus="setter"> </ Stub>

<Stub Stub-genus="getter"> </ Stub>

</ Stubs>

<LangSpecProperties>

<LangSpecProperty Key="scope" value="agent"> </ LangSpecProperty>

**Figura 3: El género agente-var-booleana tiene dos talones de cheques, un getter y setter a.**



En la actualidad hay cinco talones disponibles, cada uno con su propio género:

* + - * getter: un bloque de datos, se devuelve el valor de su bloque padre.
* setter: un bloque de comando que establece el valor de su bloque padre

<BlockFamilies>

<BlockFamily>

<FamilyMember> Fd </ FamilyMember>

<FamilyMember> Bk </ FamilyMember>

</ BlockFamily>

**Figura 4: Los géneros bloque hacia adelante y hacia atrás están en la misma familia. El bloque gráfico muestra sus familiares en su menú desplegable.**



* inc: un bloque de comandos que incrementa el valor de su bloque padre por un valor fijo.
* persona que llama: un bloque de comandos que ejecuta los comandos encerrados por su padre. Esta

stub es utilizado por bloques procedimiento para crear sus bloques de llamadas.

* agente: un bloque de datos que devuelve el valor de su bloque padre de un agente particular
  + 1. *LangSpecProperties*

La biblioteca codeblocks es independiente del lenguaje y no todas las propiedades especificadas en el

archivo de lenguaje de definición puede cubrir todas las propiedades necesarias para un lenguaje de bloques. Cada LangSpecProperty tiene una pareja clave-valor que cuando se carga se guarda en el HashMap propiedades en la clase BlockGenus.

## BlockFamilies

Familias de misión-crítica proveen mayor comodidad para el usuario por géneros bloque de agrupación

que tienen una funcionalidad similar en las familias. Cuando los bloques se representan en el espacio de trabajo, los bloques con las familias tienen una caída adicional de la celda incluida en su bloque gráfico. Los familiares se incluyen en este cuadro desplegable.

# Especificación del área de trabajo

## BlockDrawerBars y BlockDrawer

BlockDrawerBars gestionar BlockDrawers, que son contenedores de bloque. BlockDrawerBars contener una serie de botones, cada botón de cambiar la visibilidad de cajón bloque.

Hay cuatro tipos de cajones de bloques, cada uno diferente en comportamiento y funcionalidad.

* Por defecto: contiene una instancia de cada bloque de contenidos. Los bloques pueden ser arrastrados y

caer dentro del cajón y hacia y desde el bloque de lona o cajones otros bloques.

* De fábrica: contiene bloques que pueden producir una cantidad infinita de instancias de bloque.

Cuando un bloque se recogió de un cajón de fábrica, una nueva instancia se crea. Cuando un

bloque es arrastrado de nuevo a un cajón de fábrica, el bloque se elimina. Los bloques de contenidos dentro de este cajón no se movió, pero sólo producen nuevos bloques.

* Página: contiene bloques para una página en particular
  + Custom: Al igual que el cajón de bloque por defecto, excepto que puede ser este cajón guarda con el proyecto para el futuro.

|  |  |
| --- | --- |
| **Atributos BlockDrawer / elementos** | **Descripción** |
| nombre | el nombre de este cajón. El nombre se muestra en el botón correspondiente a este cajón |
| tipo | tipo cajón: default, fábrica de la página, la costumbre.  Se establece en "default" por defecto. |
| es-abierto | Si "sí", cajón se muestra en el inicio de trabajo. Se establece en "no" por defecto. |
| botón de color | El color del botón correspondiente |
| BlockGenusMember | el nombre genérico de un bloque dentro de este cajón |
| Separador | Formateo de propiedad. Si se especifica entre elementos BlockGenusMember, se traza una línea entre los miembros del género en el cajón gráfica. |
| NextLine | Formateo de propiedad. Si se especifica entre elementos BlockGenusMember, el miembro del género después del elemento nextLine se dibuja en la siguiente fila en el cajón. |

## Páginas

Páginas de dividir y organizar el espacio de trabajo de bloque. Páginas pudo haber asociado cajones de bloques que contienen bloques que son "especiales" a esa página.

|  |  |
| --- | --- |
| **Atributos de página** | **Descripción** |
| page-nombre | nombre de esta página. Nombre sea sacado de la página. |
| cajón de nombre | cajón asociadas con esta página |