Módulo de Prácticas iFP

Proyecto Web

Presentación y desarrollo del Proyecto Web

Alejandro Aibar Heras
24-5-221

Índice

Generador de bloques

En la página de generación de bloques se crearon las etiquetas para llevar a cabo una implementación de un formulario en HTML mediante la herramienta **Blockly Block Generator**[i].

Estos bloques son:

Bloque de formulario (form_block):

Tiene tres campos principales que definirán la etiqueta:

- **Name**, con el cuál se dará nombre al formulario.
- **Action**, el cuál indicará la acción del formulario (programa PHP).
- **Method**, que definirá el método del formulario (get, post, ...)

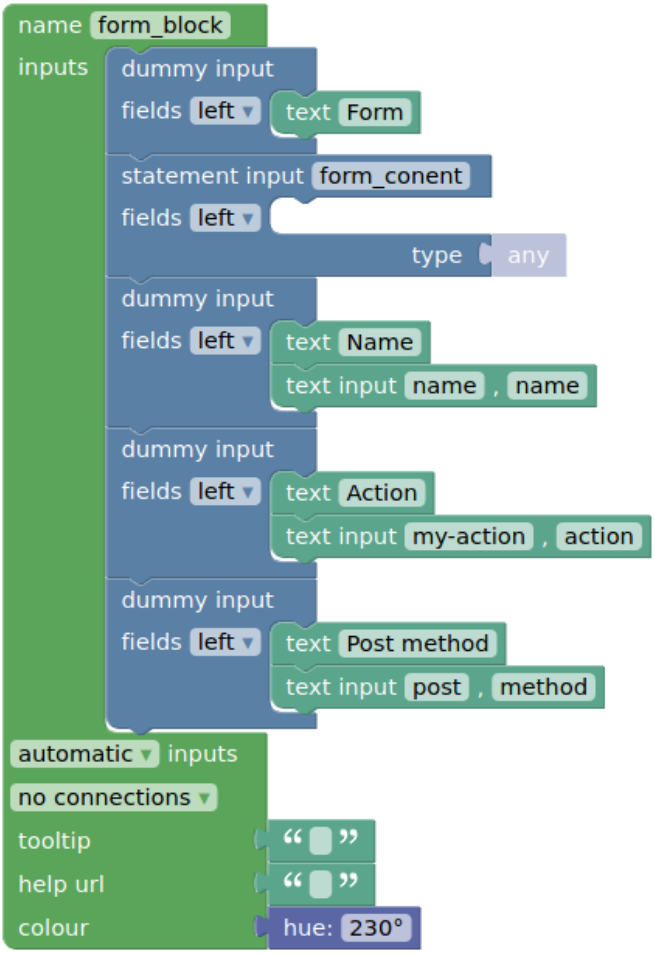


Figura 1.1
Definición del bloque FORM

Figura 1.2
Bloque final

Bloque de botón (button_block)

Contiene dos campos para definir su etiqueta:

- **Button**: Define el tipo de botón (submit, reset, button)
- **Value**: Indicará su valor, es decir, el texto que contendrá.

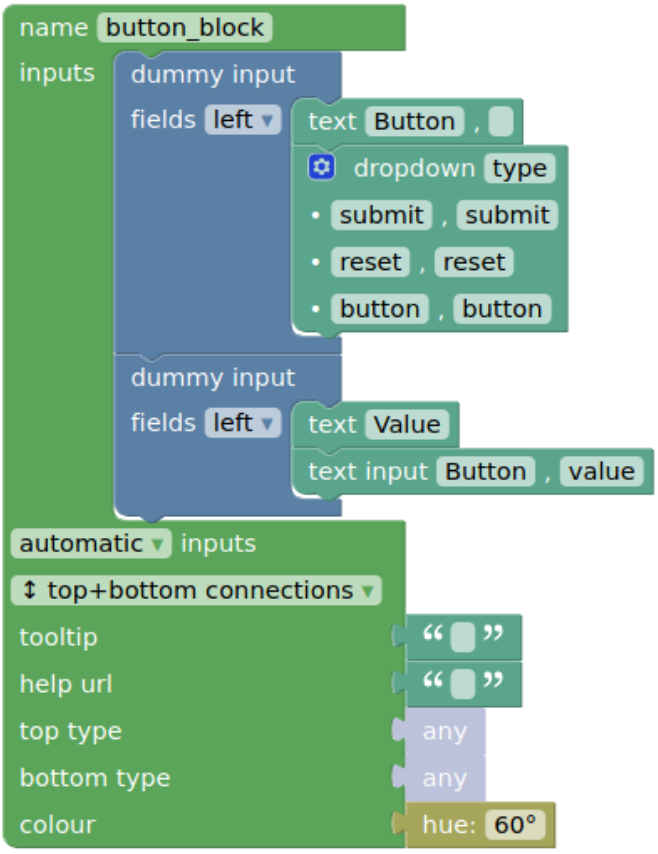


Figura 2.1
Definición del bloque BUTTON

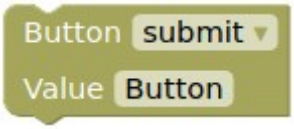
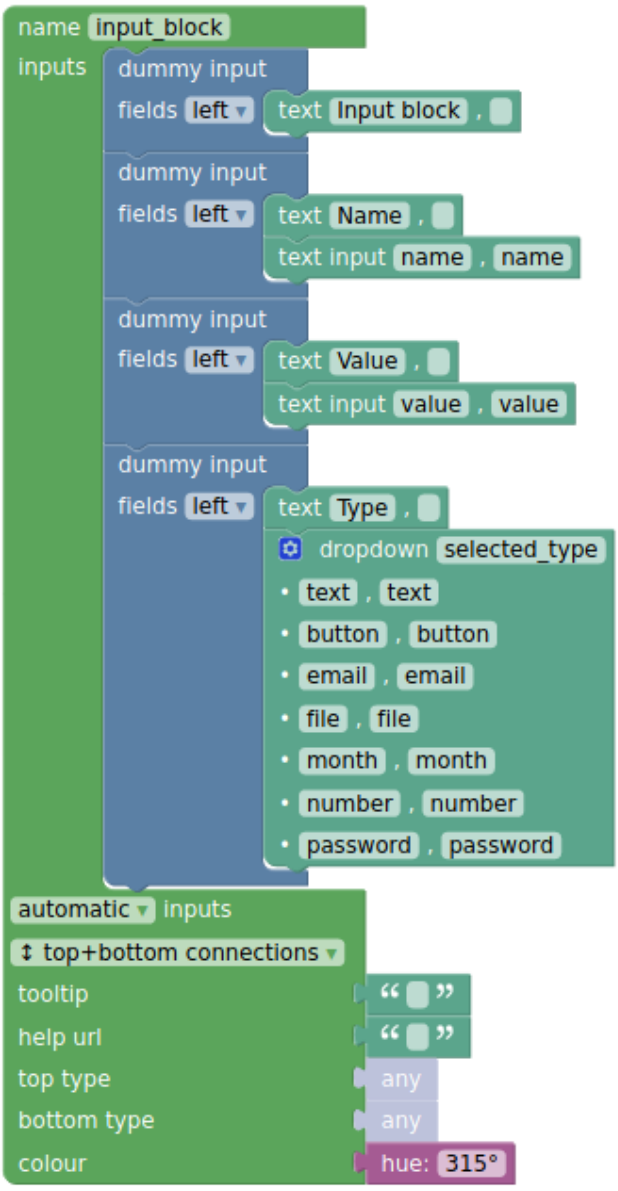


Figura 2.1
Bloque final

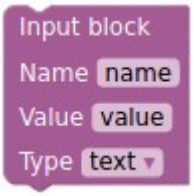
Bloque de entrada (input_block)

Posee tres campos:

- **Name**, con el cual se le podrá dar nombre a la etiqueta de entrada
- **Value**, que indicará su contenido (texto predeterminado, por ejemplo)
- **Type**, será su tipo (e-mail, file, text, ...)



Forma 3.2
Definición de bloque INPUT



Forma 3.1
Bloque final

Forma 5.1
Bloque final

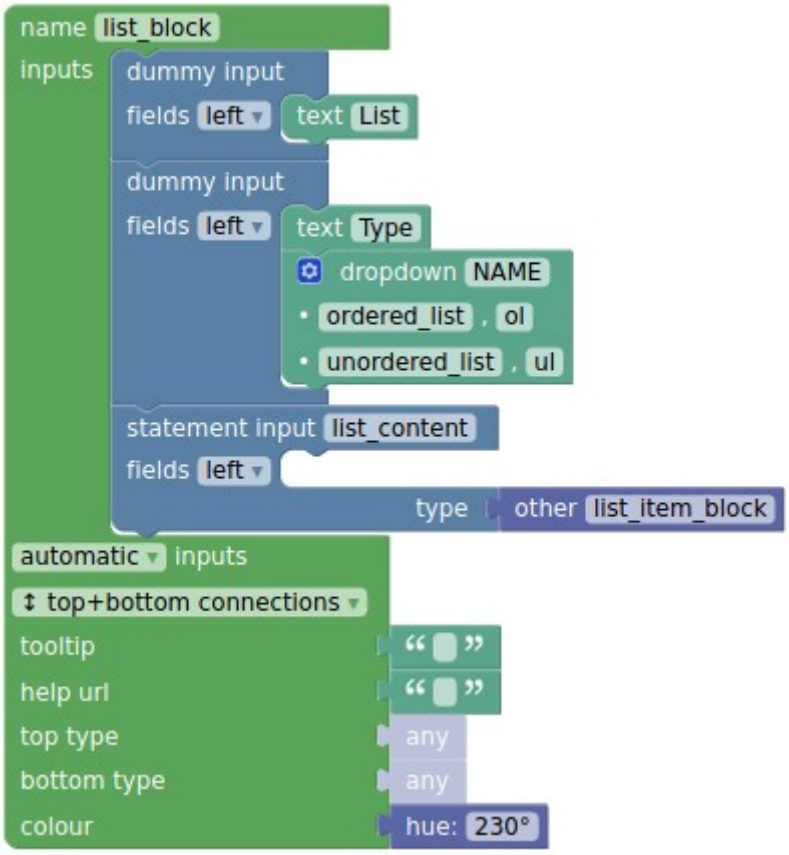
Bloque de lista (list_block)

Se compone de dos apartados:

- **Type**, que denotará el tipo de lista
- El contenido de la lista en sí, el cuál será rellenado con bloques en su interior.

El intérprete asignará una etiqueta **li** (list item) a cada objeto del contenido automáticamente, por lo que no requerirá intervención del usuario)

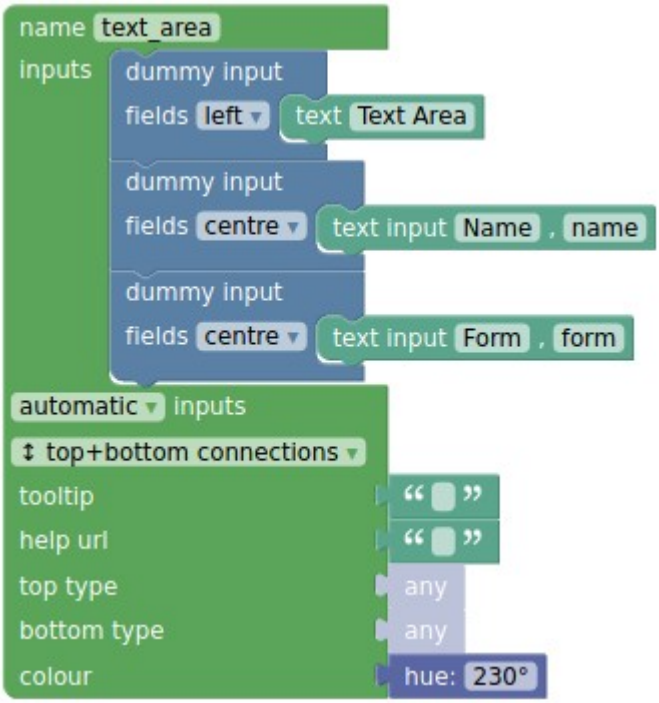
Forma 4.1
Bloque final



Forma 4.2
Definición de bloque LIST

Área de texto (text_area)

Lo componen dos etiquetas; el **nombre** que recibirá dicha área y el **formulario** al que se refiere.

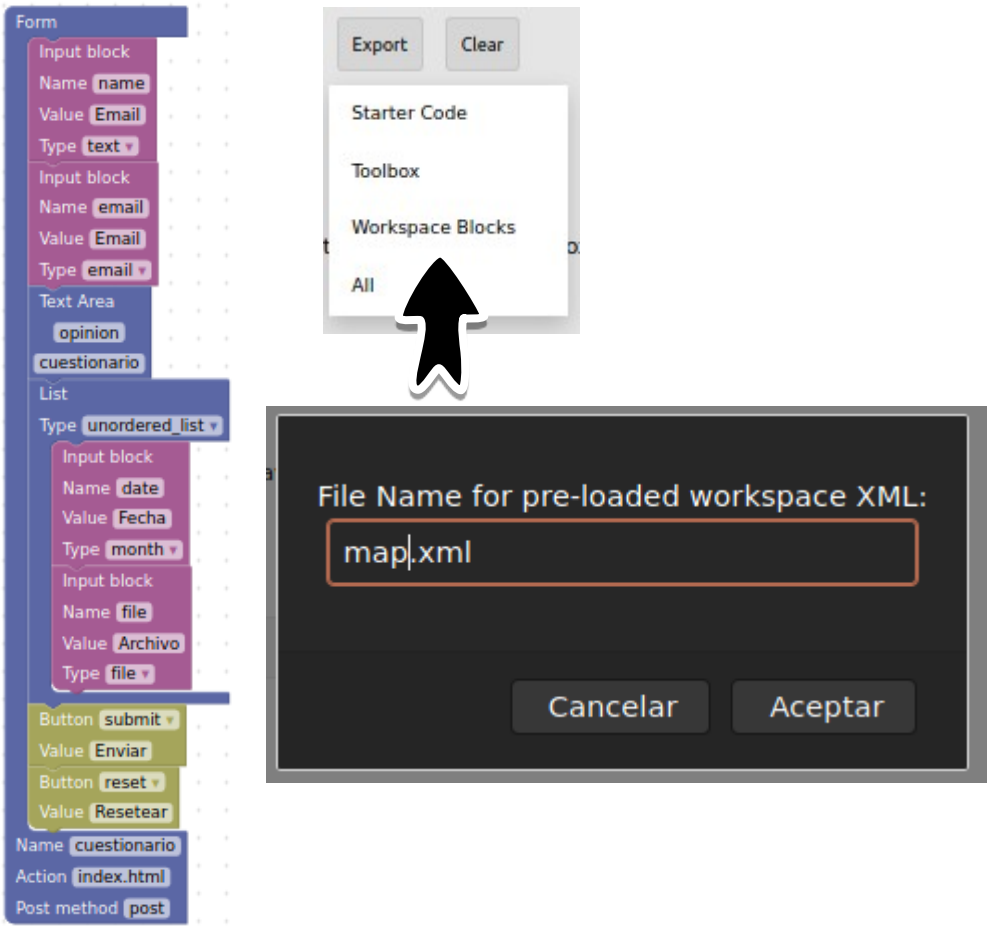


Forma 5.2
Definición de bloque TEXT AREA

Forma 5.1
Bloque final

Elaboración de bloques

La elaboración de los bloques deberá realizarse en el **Workspace Factory**, de la **Blockly Block Factory**. De este modo, al exportar el Workspace se obtendrá un archivo XML compatible con el Intérprete.



Se nombrará al archivo **map.xml**. De este modo, estará preparado para ejecutarse con el intérprete.

Programa de interpretación

El programa de interpretación, o **parser**, es el componente que se encargará de identificar los elementos del archivo XML y generar un archivo **HTML** que contenga tanto la hoja de estilos como el formulario descrito en los bloques.

Este programa necesitará 2 elementos principales:

- **El archivo XML exportado**, con nombre **map.xml**.
- **El archivo CSS con los estilos**, de nombre **styles.css**

Al finalizar la ejecución del programa se creará un archivo **index.html**. Al abrirlo en el navegador, se podrá ver el formulario generado.

El programa se ha publicado en GitHub bajo licencia de código abierto de uso libre (**GPLv3)** para su uso y redistribución bajo los términos de dicha licencia. [ii]

Existe un paquete para evitar la instalación de Python y las librerías necesarias para la ejecución, simplificando la uso del programa aún más.

Funcionamiento del programa

Se necesitarán las siguientes librerías adicionales para ejecutar el programa:

- **switchcase**, para poder ejecutar switch statements dentro del script (Python no los soporta por defecto)
- **dominate** para poder generar etiquetas HTML de forma rápida.

A grandes rasgos, la ejecución se divide en tres fases:

- **Interpretación del archivo XML**, cargando el archivo XML en memoria mediante la librería ElementTree, extrayendo posteriormente la raíz en la variable **root**, junto al *namespace* del XML.
- **Traducción de XML a HTML** (detallado a continuación)
- **Exportación del archivo**, escribiendo la información interpretada en el paso anterior a un archivo HTML

La mayor parte del programa se basa en interpretar la información del XML y traducirla a HTML. A continuación se detallan las funciones:

make_html_tag(block)

Devuelve una etiqueta HTML en base al argumento **block** (bloque del XML).

Para ello, comprueba el tipo de bloque y realiza un switch statement en el que se averigua el tipo de etiqueta que debería corresponderle.

parse_statemet(block, tag, add_element)

Se llama cuando se debe analizar un statement (es decir, el interior de un bloque)

Convierte todos los bloques a HTML que encuentre dentro de **block** y los añade a la etiqueta **tag**.

Adicionalmente se puede indicar un argumento **add_element**, que servirá para introducir a cada bloque de statement dentro de la etiqueta indicada en **add_element**, añadiendo esta misma a **tag**. Esto permite estructuras complejas como listas y tablas:

**
test
** (en este caso OL sería **tag**, y **li, add_element**)

get_statement_xml_tag(tag)

Devuelve el bloque contenido en la etiqueta **statement** dentro de la etiqueta **tag**. En caso de no hallarla, devuelve **None**.

get_next_xml_tag(tag)

Devuelve el bloque contenido en la etiqueta **next** de la etiqueta **tag**. En caso de no hallar a **next**, devuelve **None**.

namespace(element)

Es una función muy básica, utilizada únicamente para obtener el **namespace** del documento XML. El **namespace** se utiliza para realizar búsquedas dentro del árbol XML (función *get_xx_tag*) y comprobaciones adicionales (map.xml ha sido generado con Blockly, etc...)

Referencias

i Block Factory: https://blockly-demo.appspot.com/static/demos/blockfactory/index.html

ii Interprete: https://github.com/alexaib2002/blockly_xml-html_parser