Extensión del manual de usuario

Autor:
Alberto Ramos Sánchez

Agosto 2020

Índice

1. Ejecución *Addon.*

Existen dos formas de ejecutar el *Addon*: desde el código o instalándolo. Para la programación es más sencillo ejecutarlo desde el código.

Antes de ejecutarlo, debemos cambiar la ruta del código a la carpeta donde se encuentra nuestro código. Para ello, en cada fichero `__init__.py`, `exec.py` y `exec_all.py`, cambiamos la ruta de la variable `project_folder` por la ruta donde se encuentre la carpeta `blender-editor`. La ruta debe ser absoluta.

- `./blender-editor/achibuilder/__init__.py`
- `./blender-editor/archibuilder/exec.py`
- `./blender-editor/robotcontrol/__init__.py`
- `./blender-editor/robotcontrol/exec.py`
- `./blender-editor/filemanager/__init__.py`
- `./blender-editor/filemanager/exec.py`
- `./blender-editor/filemanager/__init__.py`
- `./blender-editor/filemanager/exec.py`
- `./blender-editor/exec_all.py`

Esto nos permite hacer accesible con *Python* desde el directorio donde se encuentre *Blender*, y así poder tener varios ficheros `.py` accesibles donde organizar el código.

Una vez cambiadas las rutas, abrimos *Blender* y realizamos los cambios e instalaciones explicadas en el manual de usuario menos la instalación de los *addons* propios de nuestra aplicación (Apartado 1 del apéndice A en Desarrollo de módulo de comunicación en Blender para la interoperación con plataforma robótica, 2020 Julio).

Una vez realizado los cambios debemos tener instalado en Blender el módulo *msgpack* de *Python* y activado el *addon* de Measureit.

Una vez finalizado, abrimos la pestaña *Scripting* de *Blender* (en la parte superior, figura 1). Ahí encontraremos un editor de *scripts*, con el abriremos el fichero `exec_all.py` y lo ejecutaremos. Para comenzar a utilizar el programa, vamos a la pestaña *layout*, y con la tecla N abrimos el panel derecho donde se encuentran todas las funciones. En el manual de usuario se explica como utilizarlas (Apéndice A en Desarrollo de módulo de comunicación en Blender para la interoperación con plataforma robótica, 2020 Julio).

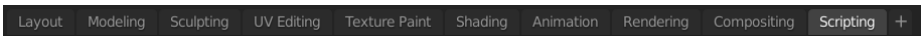

Figura 1: Workspaces.

Hay que tener en cuenta que cada vez que ejecutemos este fichero los cambios que se encuentren en la escena se sobreescribiran. Para estar seguros que no se produce ningún error extraño, es recomendable dejar vacía la escena cada vez que ejecutemos el fichero `exec_all.py`, o incluso reiniciar *Blender* cada vez que realicemos un cambio en el código.

También hay que tener en cuenta que los *scripts* que estemos ejecutando en un momento pertenecen al fichero *.blend* del proyecto que tenemos abierto. Si abrimos otro fichero *.blend* debemos volver a ejecutar `exec_all.py`.

1.1. Crear nuevos ficheros .py.

El fichero `__init__.py` es el primero que se ejecuta, tanto al instalarlo como al ejecutarlo. En este fichero se encuentran las llamadas a los métodos de registros de clases.

Para crear un nuevo fichero .py debemos importarlo en `__init__.py` y realizar una llamada a `importlib.reload(nombre_fichero)`. Si este fichero contiene funciones *autoregister* y *autounregister* con los que registrar operadores, debemos incluir también el módulo en la lista operadores. Al importar los módulos en `__init__.py`, solamente son accesibles desde un módulo los módulos que fueron importados antes.

1.2. Empaquetado de *Addon*.

Para empaquetar los addons en ficheros .zip instalables hay que realizar algunos cambios en el código.

1. Cambiar import.

En todos los ficheros debemos cambiar todos los `import` marcados entre los comentarios `# local import`. Estos `import` hacen referencia a nuestros ficheros .py, que hemos creado. Debemos cambiarlos de `import modulo` a `from . import modulo`. También debemos eliminar en los ficheros `__init__.py` el código encerrado en comentarios `# remove`.

Por este motivo, dentro de nuestro código, cualquier fichero propio que vayamos a importar debemos realizarlo por completo (de la forma `import modulo` y no `from modulo import *`).

2. Eliminar llamadas a importlib.reload en __init__.py.

Debemos eliminar todas las llamadas a `importlib.reload` en los ficheros `__init__.py`.

3. Eliminar ficheros exec.py

Los ficheros `exec.py` y `exec_all.py` no son necesarios al instalar el *addon*, por lo que podemos eliminarlos.

4. Comprimir *addons* e instalarlos.

Para crear los instalables, debemos comprimir en *zip* cada una de las carpetas que contienen ficheros `__init__.py`. Por ejemplo, *archibuilder* es un único *addon*, y para generar el instalable comprimimos la carpeta *archibuilder* en formato *zip* (la carpeta completa, no únicamente su contenido).

La instalación del *addon* se encuentra explicada en el manual de usuario (Apartado 1.2 del apéndice A en Desarrollo de módulo de comunicación en Blender para la interoperación con plataforma robótica, 2020 Julio).

2. Ventanas útiles.

1. Consola de sistema.

Consola donde se ejecuta Blender, donde se imprimen los *print* de *Python* y ciertos *report* de operadores. Para abrirla tenemos dos opciones. Si ejecutamos *Blender* desde una ventana de consola, esta será la consola del sistema de *Blender*. Por el contrario, si abrimos *Blender* con el ejecutable, la ventana de consola se abre a través de la opción en la pestaña *Window* > *Toggle System console*.

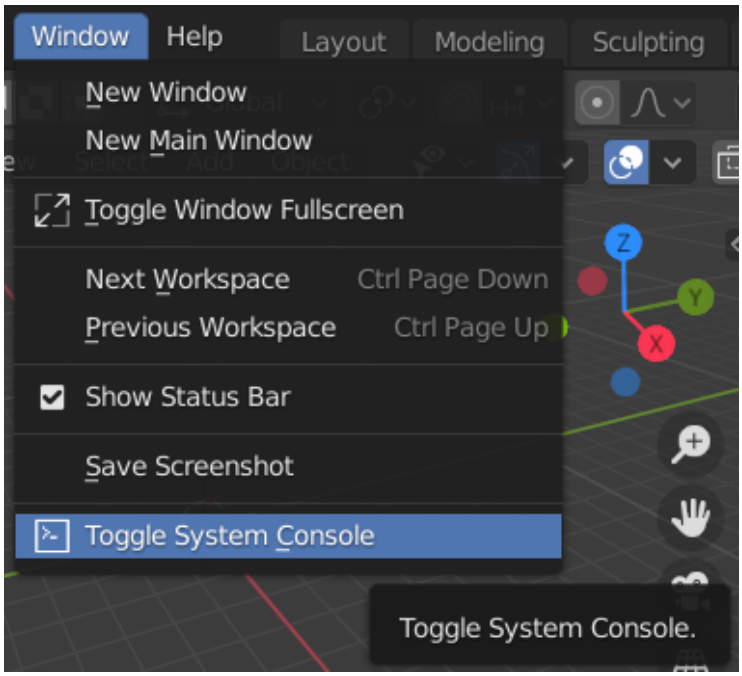

Figura 2: System console.

2. Consola *Python*

Dentro del *workspace Scripting* tenemos una consola de *Python* con la que realizar pruebas con la *api* de *Blender*.

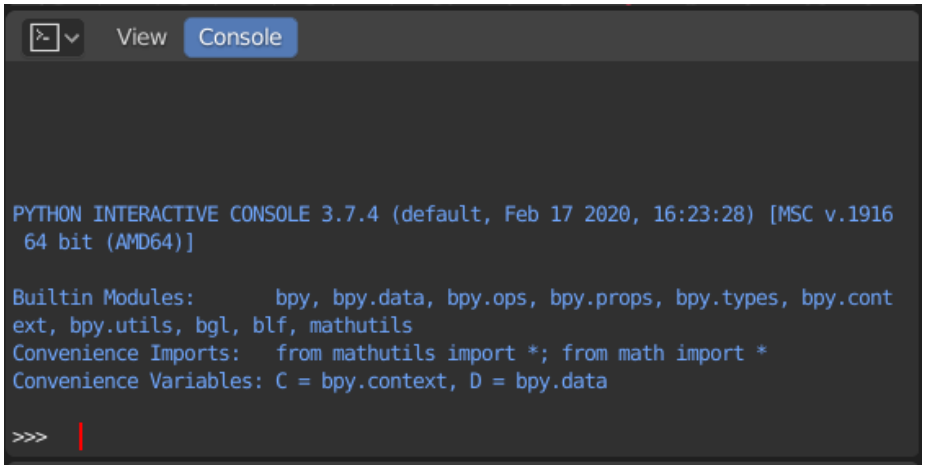

Figura 3: Consola *Python*.

3. Ventana *log*

Ventana donde se muestran todos los mensajes de los report de operadores e información acerca de operaciones que se realizan.

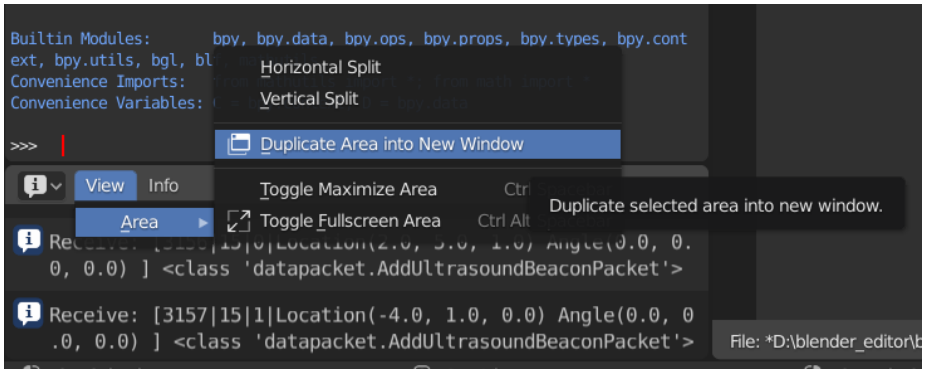

Figura 4: Ventana log.