Manual de actualización del dongle R4S con nuestros propios payloads

1.- Lo primero que debemos hacer es descargar e instalar el IDE de Arduino.

Lo podéis descargar en la siguiente web:

https://www.arduino.cc/en/Main/Software

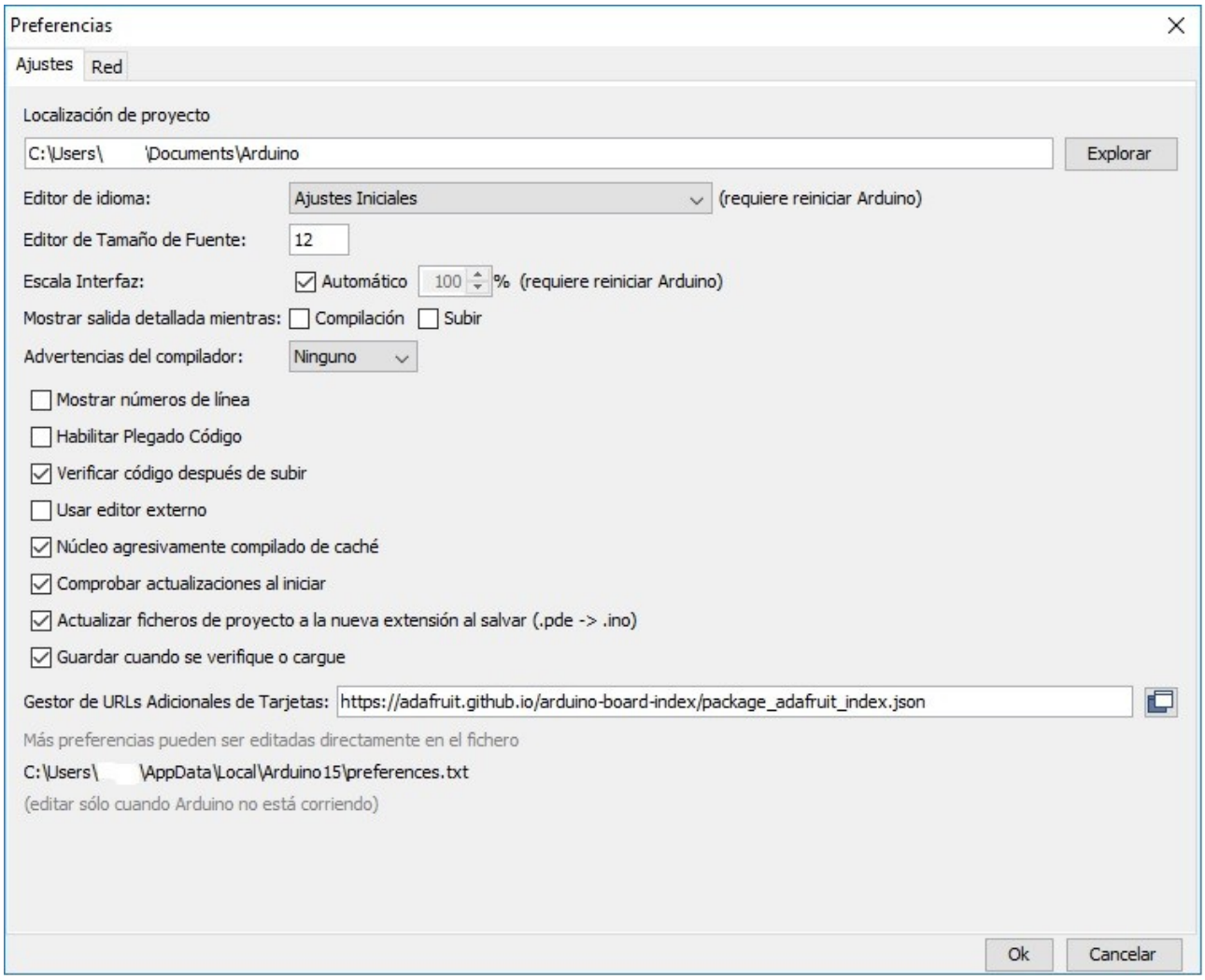
2.- Una vez instalado, descargamos el repositorio completo, lo descomprimimos, y copiamos la carpeta "fusee-launcher_for_R4S_dongle" en el directorio de proeyctos de Arduino (en Windows está en documentos/arduino).

3.- Instalamos el soporte para placas adicionales.

Para ello, abrimos el IDE de Arduino y vamos al menú Archivo -> Preferencias.

En la ventana que aparece, hay que añadir la siguiente URL en el campo "Gestor de URLs adicionales de tarjetas" y pulsar OK (si ya tenemos alguna URL añadida, hay que separarlas con una coma).

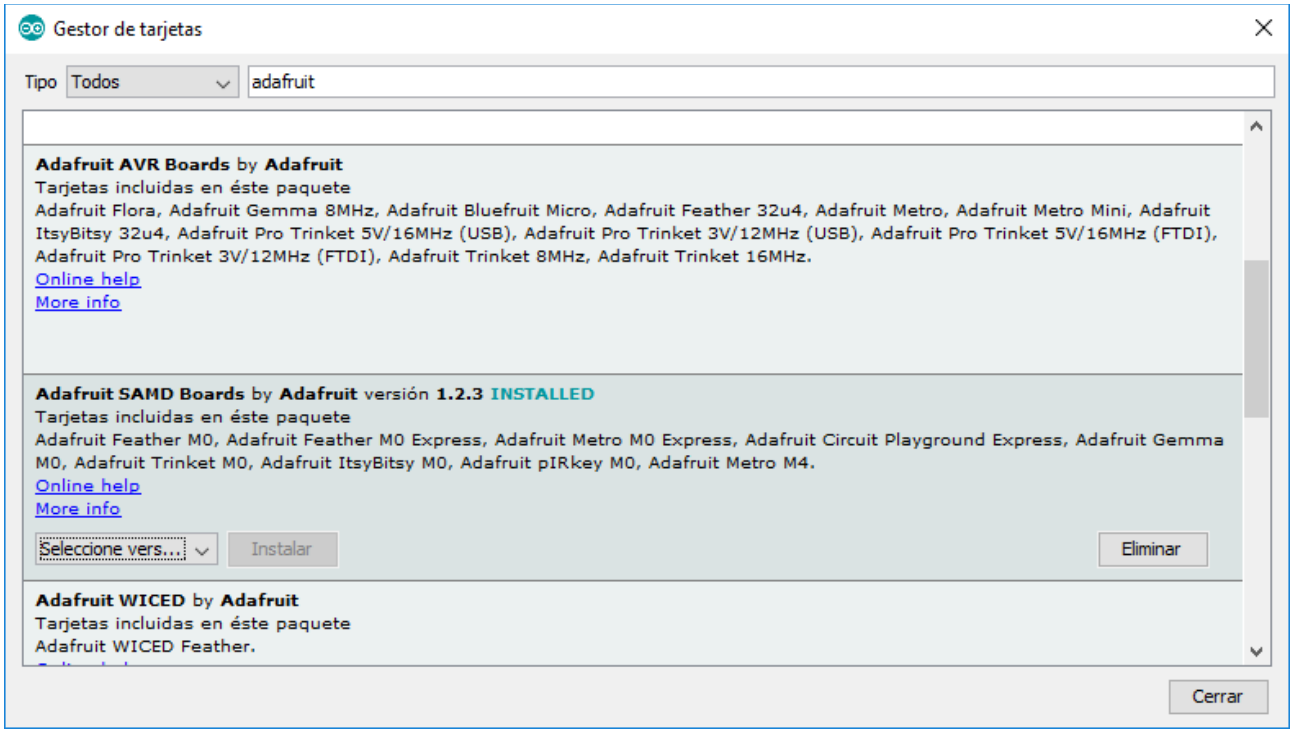
https://adafruit.github.io/arduino-board-index/package_adafruit_index.json



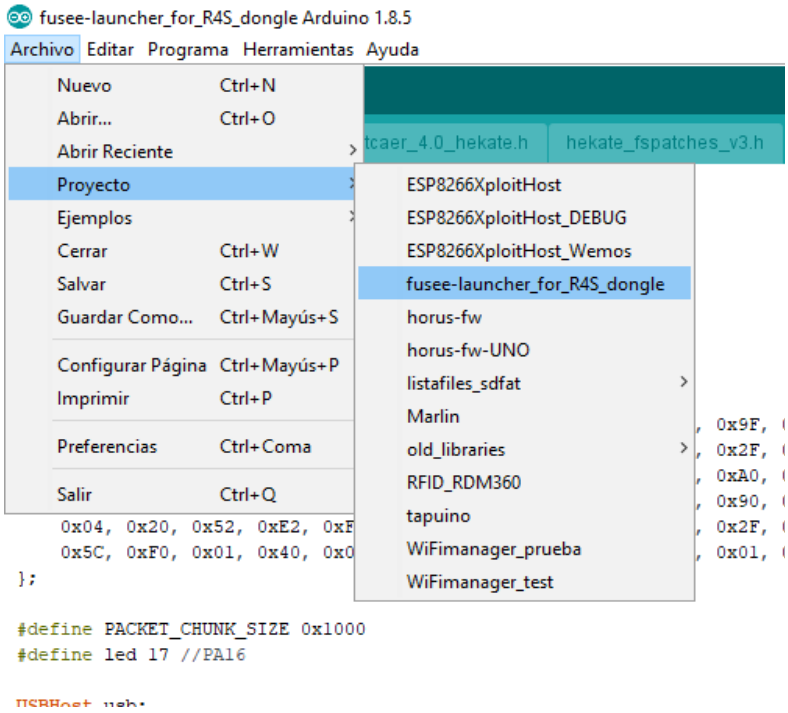
3.- Instalamos la placa para la que queremos trabajar. En este caso el Trinket M0 de Adafruit.

Para ello, vamos al menú Herramientas -> Placa -> Gestor de tarjetas

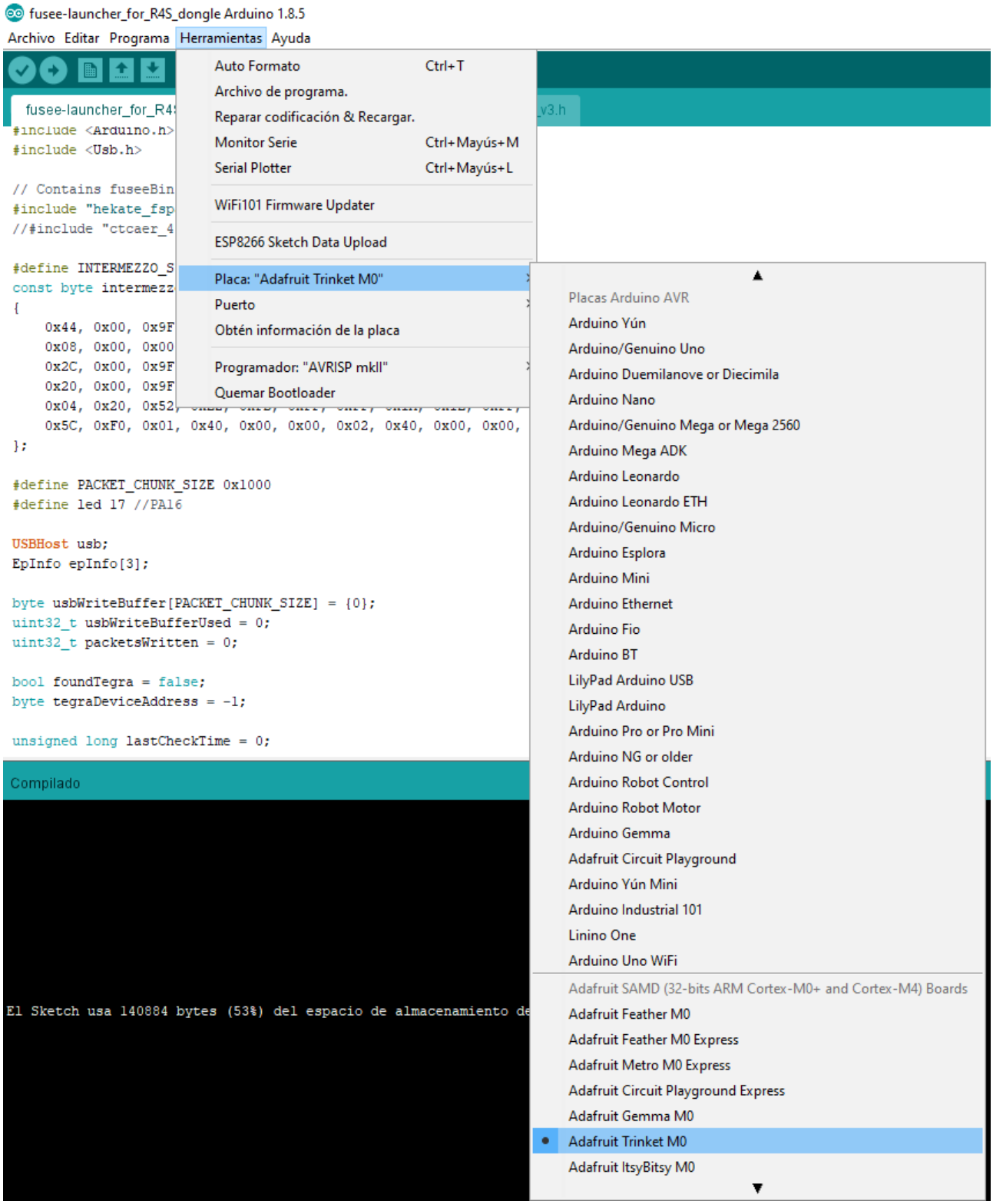
Cuando termine de descargar datos, tecleamos "adafruit" en el campo de búsqueda, y seleccionamos "Adafruit SAMD boards" en los resultados mostrados en la parte inferior. Seleccionamos la última versión (1.2.3 en el momento de escribir este manual), y pulsamos instalar.



4.- Ahora ya podemos abrir el proyecto que hemos copiado previamente desde el menú Archivo -> Proyecto.



Una vez abierto el proyecto, y antes de poder compilar, debemos seleccionar la placa para la que vamos a compilar. Para ello vamos al menú Herramientas -> Placa y seleccionamos Adafruit Trinket M0



El proyecto tiene 2 ficheros: un .ino que es el código principal, y un .h que es el payload que estamos incluyendo, y que corresponde con los payloads en .bin que nos ofrecen los desarrolladores y que enviamos cuando entramos en modo RCM. Ese fichero .bin que descargamos (siempre de los repos de los desarrolladores!) hay que convertirlo en .h mediante

la aplicación de Python "binconverter.py" incluida en el directorio tools de mi repositorio. Yo lo estoy ejecutando desde Python 2.7 en Ubuntu, creo que no punfionan en Python 3.x

Para convertir vuestro fichero .bin en .h hay que ejecutar el siguiente comando :

```
python binconverter.py fichero_payload.bin
```

Esto nos generará un archivo llamado fichero_payload.h que debemos copiar a nuestro directorio de proyecto en Arduino y modificar la siguiente línea en el fichero .ino para incluir el nombre correcto del fichero .h

```
#include "ctcaer_4.0_hekate.h"
```

Si queréis que el IDE de Arduino os muestra correctamente los ficheros nuevos que habéis copiado en el directorio, hay que cerrarlo y volver a abrir el proeycto. Si simplemente queréis compilar, y habéis escrito el nombre del fichero bien en el #include, no hace falta.

Una vez hechos los cambios, vamos a compilar para obtener el código compilado, y poder convertirlo a formato UF2 que es lo que necesitamos para poder metérselo al dongle.

Para ello vamos al menú Programa -> Exportar binarios compilados

Esto nos generará un fichero .bin (fusee-launcher_for_R4S_dongle.ino.trinket_m0.bin) en el directorio de proyecto de Arduino. Ese es el fichero que vamos a convertir al formato UF2.

Para convertirlo en UF2, simplemente ejecutar la aplicación uf2conv.py incluida en el directorio tools del repositorio de esta forma:

```
python uf2conv.py -c -f SAMD21 -o CURRENT.UF2 fusee-launcher_for_R4S_dongle.ino.trinket_m0.bin
```

Esto nos generará el fichero CURRENT.UF2 que ya podemos copiar al dongle.