

INTRODUCCION.

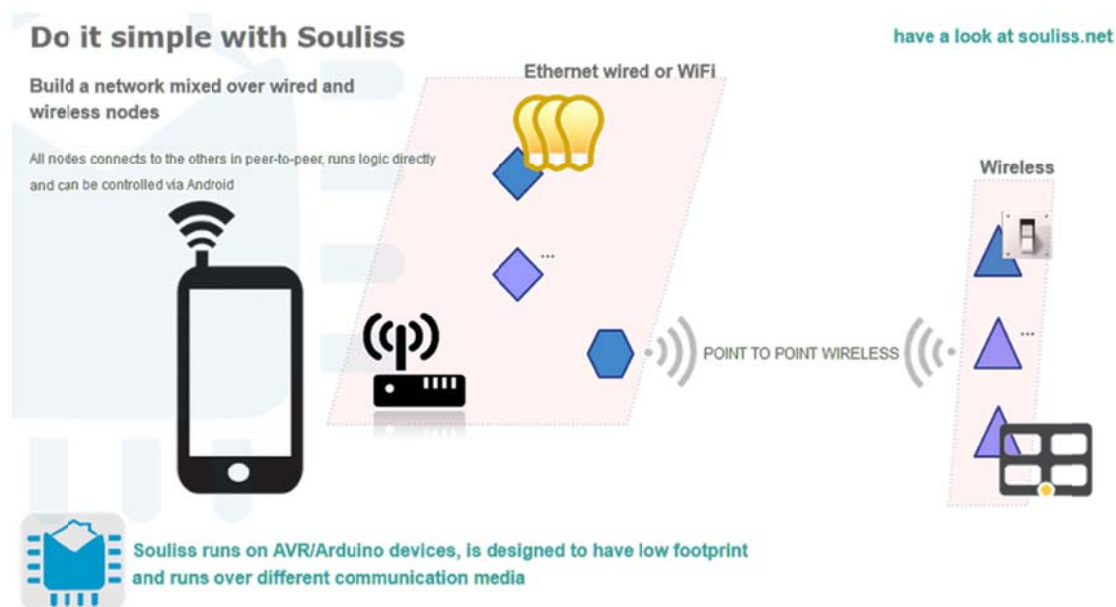
Esta documentación contiene toda la información para configurar la red de Souliss, incluye detalles sobre las arquitecturas e implementaciones soportadas.

DESCRIPCION BREVE DE SOULISS.

Souliss está construido sobre tres capas que constituye una completa red de nodos con lógica y funcionalidad distribuidas, todos los nodos pueden intercambiar datos peer-to-peer y no es necesario que un nodo central coordine las funciones lógicas y las comunicaciones.

Si usted tiene una placa compatible puede empezar con cualquiera de los ejemplos.

Gracias a su estructura escalable, la funcionalidad de Souliss se puede mover con diferentes nodos o fusionarlo en uno solo la mejor solución puede ser orientada con el tamaño de la red y sus requerimientos.



CAPAS SOULISS

Una instalación flexible es algo muy sencillo gracias a la construcción de la aplicación por capas. Desde el punto de vista del usuario estas capas no se utilizan directamente, porque están incrustadas en la API de Souliss.

vNet

La comunicación se establece a través de vNet, ella construye una red virtual para llevar a cabo los complementos y el enrutamiento a través de diferentes medios de comunicación (por cable e inalámbrica) a un nivel inferior. Sin requerir ningún tipo de configuración especial en su propia red.

vNet incluye controladores para muchos transceptores y funciona sobre todos los controladores de los medios de comunicación, incluso si no tiene un mecanismo de detección de colisiones

MaCaco

Es un protocolo sin estado y basado en eventos, es binario y pequeño y permite la comunicación punto a punto entre los nodos. Se implementa también en la aplicación Android dando una conexión directa, el mecanismo basado en eventos ahorra batería y ofrece una rápida interacción.

Lógicas e interfaces

El manejo de luces, ventanas, puertas, etc están constituidas de manera típica, un conjunto de lógicas preconfiguradas que también tienen una interface disponible en la aplicación Android. Todos los nodos pueden proporcionar una lista de estados utilizados en ella. Leer la página de [Estructura de datos](#) para saber más.

Los datos de la red Souliss se pueden recuperar mediante interfaces, una colección de protocolos para la transferencia de datos diseñados para integrar Souliss con otros dispositivos o interfaces de usuarios. Como por ejemplo, se puede utilizar la interfaz HTTP/XML para conectar su red Souliss a openHAB.

Hardware Souliss

Hay varias plataformas disponibles que se pueden ejecutar en Souliss, todas están basadas en microcontroladores AVR y tienen diferentes transceptores de extensiones de E/S. Eche un vistazo a la [Plataforma de Hardware Soportado](#).

PRIMEROS PASOS CON SOULISS.

Souliss es una estructura que funciona con diferentes configuraciones de hardware, por lo que es necesario una pequeña configuración antes de compilar, la siguiente lista le ayudara en el primer contacto con Souliss. Tómese su tiempo para leer la configuración básica, esto le ayudara a ahorrar tiempo en la construcción y puesta en marcha de su primer proyecto.

Primeros pasos:

- Descargue la última versión del IDE de Arduino.
- Verifique la compatibilidad del hardware.
- Descargue el ultimo código de Souliss para Arduino / AVR.
- Descargue la última aplicación de Souliss para Android.
- Verifique el ejemplo Blink (parpadeo).
- Configure las funciones de Souliss.
- Compile y cargue el Sketch de Souliss y... ¡¡A disfrutar!!

PASO 1: Descargue la última versión del IDE de Arduino.

Souliss está escrito principalmente en C, con un poco de C++ y puede ser fácilmente adaptado a cada microcontrolador, aunque está publicado para ser compilado directamente con el Arduino IDE.

En las notas publicadas se especifica cual la última versión soportada del Arduino IDE, generalmente usted puede usar el ultimo IDE en la versión 1.

PASO 2: Verifique la compatibilidad del hardware.

Souliss incluye drivers a bajo nivel para la interacción con el hardware transmisor y las placas E/S, por lo que se puede ejecutar de forma sencilla sobre la plataforma seleccionada. La WIKI contiene una lista del [hardware compatible](#) que se puede usar.

Todas las placas compatibles con Arduino pueden funcionar con Souliss, pero no todas las plataformas pueden ejecutar el conjunto completo de funcionalidades, como Souliss está escrito para caber en un ATmega328P y ocupa menos de 2KBytes de memoria RAM.

Si está planeando usar una placa Arduino que no aparece en listado de arriba, debe tener cuidado de usar una placa con un controlador de red compatible como **ENC28J60, W5100 o AT86RF230**.

PASO 3: Descargue el ultimo código de Souliss para Arduino / AVR.

Desde la página [web](#) usted puede descargar de forma comprimida el último código de Souliss, la aplicación Android oficial también es muy recomendada para el inicio.

PASO 4: Descargue la última aplicación de Souliss para Android.

Hay múltiples interfaces que pueden ser utilizadas con Souliss, la aplicación [Android](#) oficial es la más recomendable para empezar.

Usted puede descargar la última aplicación de Souliss para Android desde [Google Play Store](#) y instalarla en su dispositivo.

La configuración básica solo requiere la dirección local de Souliss, Comience con su red local LAN y planea configuraciones más complejas y seguras (reenvío de puertos, VPN).

El uso en segundo plano de la [App](#) permite mantener la aplicación sincronizada con la red de Souliss. Por defecto viene desactivada pero le recomendamos que la active con el fin de disfrutar de todas sus características.

PASO 5: Verifique el ejemplo Blink (parpadeo).

Si es la primera vez que usa su placa de Arduino, la mejor forma de empezar es con un ejemplo sencillo, para estar seguro que su IDE está listo para compilar. En el menú → Archivo → Ejemplos → Basics cargue el Sketch de Blink.

Desde la página web de Arduino (<http://www.arduino.cc/>) se puede obtener una guía de inicio para el ejemplo Blink. Una vez compilado ejecutado el ejemplo Blink de forma correcta, puede pasar al siguiente paso.

PASO 6: Configure las funciones de Souliss.

Es necesario configurar cual es el Hardware que se va a utilizar, siga la [Guía de configuración rápida](#) para configurar correctamente Souliss antes de su compilación. Luego asegurese que tiene la configuración adecuada de la IP con respecto a su red.

PASO 7: Compile y cargue el Sketch de Souliss y... ¡¡A disfrutar!!

Una vez configuradas las reglas generales, cargue el ejemplo [Souliss_ex01>HelloWorld_eth1](#) del IDE en Archivo → Ejemplos → Souliss y después compílelo.

Después empiece jugar con la aplicación, configure la dirección IP de tu placa Souliss e interactúe con las cosas.

Disfrute y use el foro si necesita más apoyo.

CONFIGURACION RAPIDA

La configuración rápida es una forma simplificada para la configuración de la estructura, básicamente permiten especificar la configuración de hardware y luego cargar los controladores correspondientes y el código.

Como primera opción, puede configurar su configuración directamente en el archivo `conf/QuickCfg.h` antes de compilar y cargar el código en la placa. Pero es más cómodo definir los parámetros directamente [en el sketch](#), de manera que usted no tiene que modificar el archivo `QuickCfg.h` cada vez que se compila un nodo.

HABILITAR LA CONFIGURACION RAPIDA

```
/*
*****
*/
/*!
Habilitar la configuración rápida, si está habilitada se descartarán los
parámetros de configuración de otros archivos y crea la configuración
únicamente de acuerdo con los detalles insertados en este archivo.

Si esta desactivada, la configuración será la detallada en el Modo de
Configuración.

Value      Media
0x00       Desactivada (Por defecto)
0x01       Activada
*/
*****
#define QC_ENABLE 0x01
```

La configuración rápida está habilitada de forma predeterminada, pero no es la única manera de configurar Soulliss. No vamos a dar detalles aquí sobre los modos de configuración avanzada, ya que con Configuración Rápida cubriremos casi todos los casos.

El siguiente paso es la selección del hardware, en el archivo hay un listado de la configuración compatible y es sólo necesario para insertar el valor correspondiente a la `QC_BOARDTYPE` a definir.

SELECCIONAR EL TIPO DE PLACA

```
/*
*****
*/
/*!
Seleccione el tipo de placa y la interface de conexión (wireless,
ethernet o Wi Fi), usando esta opción no se requiere ninguna otra configuración
para los nodos estándares.

La configuración estándar siempre es posible.

Value
0x00 No seleccionado (Por defecto)
0x01 Freaklabs Chibiduino (2.4 GHz Wireless)
0x02 Freaklabs Chibiduino with Ethernet Shield (W5100)
0x03 Arduino Ethernet (W5100)
0x04 Arduino with Ethernet Shield (W5100)
0x05 Arduino with ENC28J60 Ethernet Shield
0x06 KMTronic DINO v1
0x07 Olimex AVR-T32U4 with MOD-ENC28J60 (UEXT)
0x08 Olimex OLIMEXINO-32U4 with MOD-ENC28J60 (UEXT)
0x09 Olimex OLIMEXINO-328 with MOD-ENC28J60 (UEXT)
0x0A Olimex AVR-T32U4 with MOD-WIFI (UEXT)
0x0B Olimex OLIMEXINO-32U4 with MOD-WIFI (UEXT)
0x0C Olimex OLIMEXINO-328 with MOD-WIFI (UEXT)
0x0D Olimex AVR-T32U4 with MOD-ENC28J60 and MOD-I/O (UEXT)
0x0E Olimex OLIMEXINO-32U4 with MOD-ENC28J60 and MOD-I/O (UEXT)
```

```

0x0F Olimex OLIMEXINO-328 with MOD-ENC28J60 and MOD-I/O (UEXT)
0x10 Olimex AVR-T32U4 with MOD-WIFI and MOD-I/O (UEXT)
0x11 Olimex OLIMEXINO-32U4 with MOD-WIFI and MOD-I/O (UEXT)
0x12 Olimex OLIMEXINO-328 with MOD-WIFI and MOD-I/O (UEXT)
0x13 Olimex AVR-T32U4 with MOD-ENC28J60 and MOD-I/O 2(UEXT)
0x14 Olimex OLIMEXINO-32U4 with MOD-ENC28J60 and MOD-I/O 2(UEXT)
0x15 Olimex OLIMEXINO-328 with MOD-ENC28J60 and MOD-I/O 2(UEXT)
0x16 Olimex AVR-T32U4 with MOD-WIFI and MOD-I/O 2(UEXT)
0x17 Olimex OLIMEXINO-32U4 with MOD-WIFI and MOD-I/O 2(UEXT)
0x18 Olimex OLIMEXINO-328 with MOD-WIFI and MOD-I/O2(UEXT)
0x19 Olimex AVR-T32U4 with MOD-ENC28J60 and MOD-RGB(UEXT)
0x1A Olimex OLIMEXINO-32U4 with MOD-ENC28J60 and MOD-RGB(UEXT)
0x1B Olimex OLIMEXINO-328 with MOD-ENC28J60 and MOD-RGB(UEXT)
0x1C Olimex AVR-T32U4 with MOD-WIFI and MOD-RGB(UEXT)
0x1D Olimex OLIMEXINO-32U4 with MOD-WIFI and MOD-RGB(UEXT)
0x1E Olimex OLIMEXINO-328 with MOD-WIFI and MOD-RGB(UEXT)
0x20 KMP Electronics DINO v2
0x30 DFRobots XBoard Relay
0x31 DFRobots XBoard
0x40 Freaklabs Chibidino with ENC28J60 Ethernet Shield
0x41 Arduino Ethernet (or Ethernet Shield) with USART
0x42 Arduino with ENC28J60 Ethernet Shield and USART
0x43 Arduino with USART
*/
/*****/
#i f(QC_ENABLE)
#       define    QC_BOARDTYPE                0x00
#endi f

```

Un ejemplo, si usted está planeando usar Arduino con Ethernet Shield (W5100) usted puede editar el archivo incluyendo **0x04**.

```

#i f(QC_ENABLE)
#       define    QC_BOARDTYPE                0x04
#endi f

```

A partir de esta paso la infraestructura conoce cuales son los controladores necesarios a utilizar, por eso debemos de especificar si el nodo será utilizado como Gateway.

ACTIVANDO EL GATEWAY (SI ES NECESARIO).

```

/*****/
/*!
Seleccionar el tipo de Gateway solicitado (si es necesario), un nodo Gateway
recoge los datos de todos los demás en la red y hacer lo mismo con los
interfaces externos disponibles.

Value
0x00    No selection (Default)
0x01    Gateway
0x02    Gateway and Data Persistence
0x03    Gateway with Arduino Ethernet Library Compatibility
0x04    Gateway with Arduino Ethernet Library Comp. and Data Persistence
*/
/*****/
#i f(QC_ENABLE)
#       define    QC_GATEWAYTYPE              0x00
#endi f

```

Un nodo Gateway puede recopilar datos de todos los demás en la red y los pone a disposición de todas las interfaces de usuario compatibles, mediante la selección por defecto. El Gateway and data se hará efectiva sólo a través de Macaco.

Si desea transferir datos utilizando una biblioteca externa basado en Arduino Ethernet Library debe ser seleccionado el Gateway.

El modo Persistence puede activarse, esto permite a un solo nodo recuperar los datos de todos los demás en la red. Consulte este enlace para obtener límites.

HABILITAR UNA INTERFACE (SI ES NECESARIO)

```
/*
*****
*/
/*!
Seleccione las interfaces de recuperación de datos, que puedan ser utilizados
ya sea en el Gateway o nodos.

Value
0x00    No selection (Default)
0x01    HTTP Command Parser
0x02    openHAB HTTP XML Interface

*****
*/
#if (QC_ENABLE && !defined(INTERFACE_INSKETCH))
#define QC_INTERFACE 0x00
#endif
```

Las interfaces son parte del código que hace que los datos disponibles a través de diferentes tipos de protocolo, esto hace fácil integrar Souliss con interfaces de usuario externo.

NODOS CON ETHERNET, CONFIGURACION IP

```
/*
*****
*/
/*!
Configuración de IP

La dirección IP de tarjetas Ethernet se define como la combinación de una base
de direcciones IP y la dirección vNet, para conseguir esto el
DEFAULT_BASEIPADDRESS [] no contendrá los bits que son cero en la máscara de
subred, los últimos bits se establecen con la dirección vNet.

A continuación se enumeran algunos ejemplos de configuraciones válidas y no
válidas, la configuración por defecto coincide con la utilizada por la mayoría
de las redes domésticas routers.

Example of valid configuration are:
- IP 192.168. 0.0 / SUBNETMASK 255.255.255.0
- IP 192.168. 1.0 / SUBNETMASK 255.255.255.0
- IP 192.168.10.0 / SUBNETMASK 255.255.255.0
- IP 192.168. 0.0 / SUBNETMASK 255.255. 0.0
Example of wrong configuration are:
- IP 192.168. 0.12 / SUBNETMASK 255.255.255.0 (WRONG)
- IP 192.168. 10.0 / SUBNETMASK 255.255. 0.0 (WRONG)

*/
*****
*/
#if (QC_ENABLE)
const uint8_t DEFAULT_BASEIPADDRESS[] = {192, 168, 1, 0};
const uint8_t DEFAULT_SUBMASK[] = {255, 255, 255, 0};
const uint8_t DEFAULT_GATEWAY[] = {192, 168, 1, 1};
#endif
```

Todos los nodos en las redes Souliss tiene una dirección vNet, esta dirección es de dos bytes, donde el primer byte definen los medios de comunicación y la otra el número de nodo. Esta dirección vNet necesita ser traducido a una dirección IP si se quiere transmitir sus datos vNet a través de una red Ethernet/IP.

Es necesario mover vNet sobre IP si desea comunicarse con la aplicación para Android y/o con un master Modbus TCP, podría que se necesitare recibir o enviar información a otros nodos a vía IP (podría hacerse también si ella). Básicamente todo la instalación de Soulliss necesita una configuración IP.

El uso de vNet sobre IP está diseñado para ser en una LAN (o más de una VPN) así que los datos que se especifican son válidos para todos los nodos. Para comprender cómo realizar esta configuración, usted necesita saber que la dirección IP de un nodo Ethernet/IP Soulliss está relacionada con la dirección de vNet del propio nodo. Por lo tanto, es necesario especificar sólo parcialmente la dirección IP.

Vamos a hacer un ejemplo, usted tiene un nodo en el que la dirección de vNet es 0x0011 (que es decimal 17) y su dirección del router es 192.168.1.1, con máscara de subred 255.255.255.0

```
#define network_address_1      0x0011      // 0x0011 igual a 17 en decimal
```

Usted necesita especificar sólo la red de tu router como DEFAULT_BASEIPADDRESS para Soulliss, entonces 192.168.1.0, esa es la dirección IP del router filtrado por el mascara de subred, por lo tanto:

```
#if(QC_ENABLE)
const uint8_t DEFAULT_BASEIPADDRESS[] = {192, 168, 1, 0};
const uint8_t DEFAULT_SUBMASK[] = {255, 255, 255, 0};
const uint8_t DEFAULT_GATEWAY[] = {192, 168, 1, 1};
#endif
```

La dirección IP de su dispositivo Soulliss será la suma de la BAEIPADDRESS y la suma de la dirección vNet en el último byte, dando como resultado 192.168.1.17. Usted debe ser capaz de hacer ping a él, eso significa que la configuración funciona correctamente.

La dirección MAC se ajusta automáticamente a una administración local, es necesario hacer una configuración detallada si usted desea hacerlo de forma diferente.

UTILICE UNA CONFIGURACION IP ESTATICA O DHCP

La configuración IP como se describe anteriormente da al usuario un control total de la configuración vNet, pero para un enfoque más sencillo incluso se puede configurar una dirección IP directamente en su sketch.

Antes de que el setup() se incluirá lo siguiente para definir los parámetros de configuración IP:

```
// Defina la configuración de la red
uint8_t ip_address[4] = {192, 168, 1, 17};
uint8_t subnet_mask[4] = {255, 255, 255, 0};
uint8_t ip_gateway[4] = {192, 168, 1, 1};
```

Dentro del setup() utilice los parámetros:

```
Soulliss_SetIPAddress(ip_address, subnet_mask, ip_gateway);
SetAsGateway((U16)ip_address[3]); // Utilice 17 como Dirección vNet
```


Alternativamente, puede utilizar la librería estándar de Arduino Ethernet para definir una dirección IP o utilizar DHCP y DNS, siga guía de la [librería de Arduino Ethernet](#) como referencia.

TU PRIMER PROGRAMA

INTRODUCCION

Con el fin de simplificar la primera subida de los Sketch de Souliss en tu Arduino o placa compatible, hemos construido una guía paso a paso con las capturas de pantalla correspondientes. Está basado en Windows, pero es similar para las personas que utilizan Arduino IDE en Linux.

REQUISITOS PREVIOS

Ser usuario de Arduino y haber sido capaz de cargar el ejemplo Blink (Parpadeo).

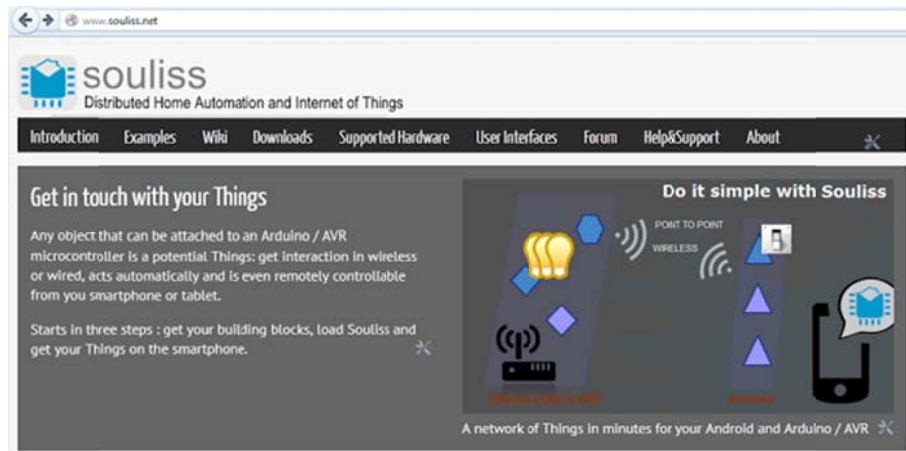
VAMOS A EMPEZAR

En el momento de la creación de esta guía la Arduino IDE usada a sido la 1.0.3 y la última versión Souliss es el Alfa 4.5, utilice la más reciente disponible en el momento.

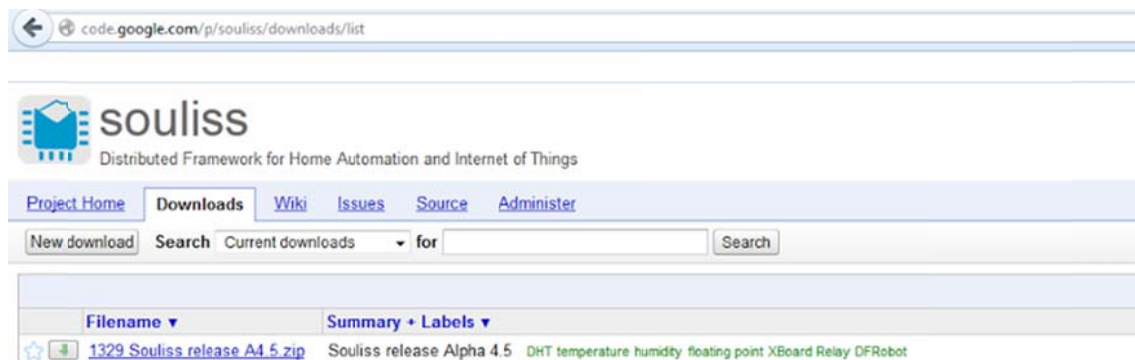
En esta guía se mostrará cómo cargar boceto HelloWorld la Souliss ', no se muestra aquí cómo configurar la aplicación para Android SoulissApp.

OBTENER EL ULTIMO CODIGO SOULISS

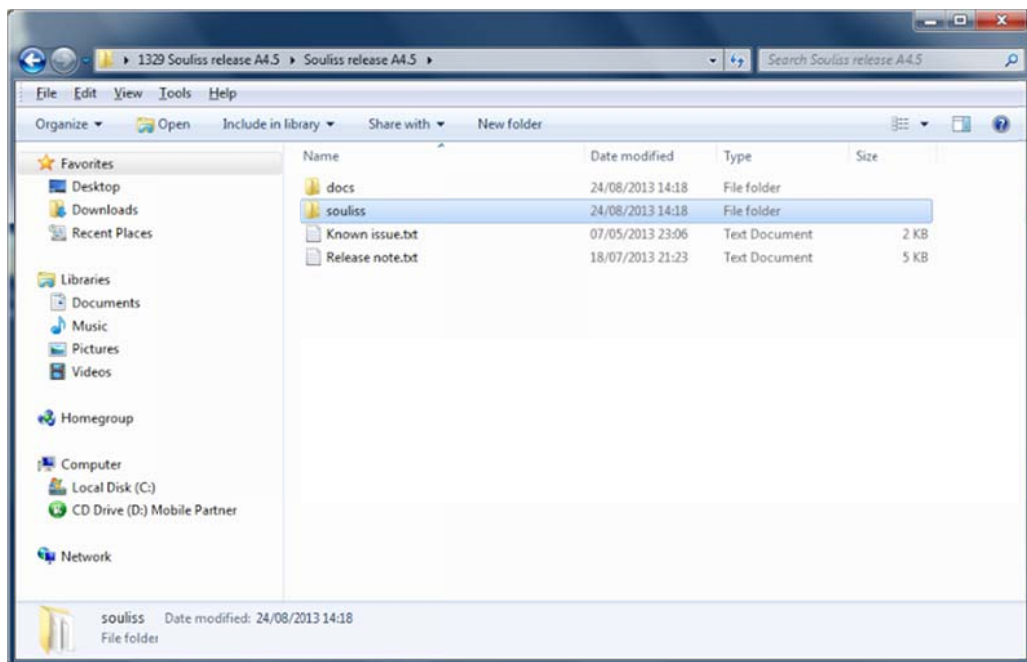
Usted puede obtener el último código Souliss de [Google Code Repository](https://code.google.com/p/souliss/), acceder al desde el enlace anterior o desde la página web Souliss.



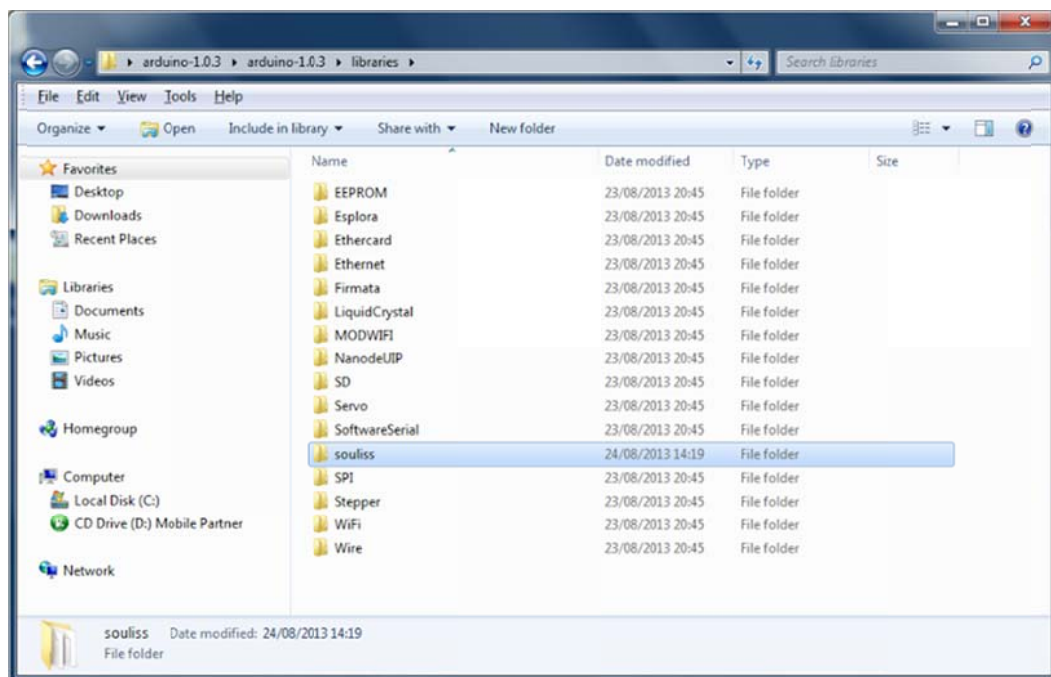
Ir a la página de descargas y seleccionar el enlace de Google Drive.



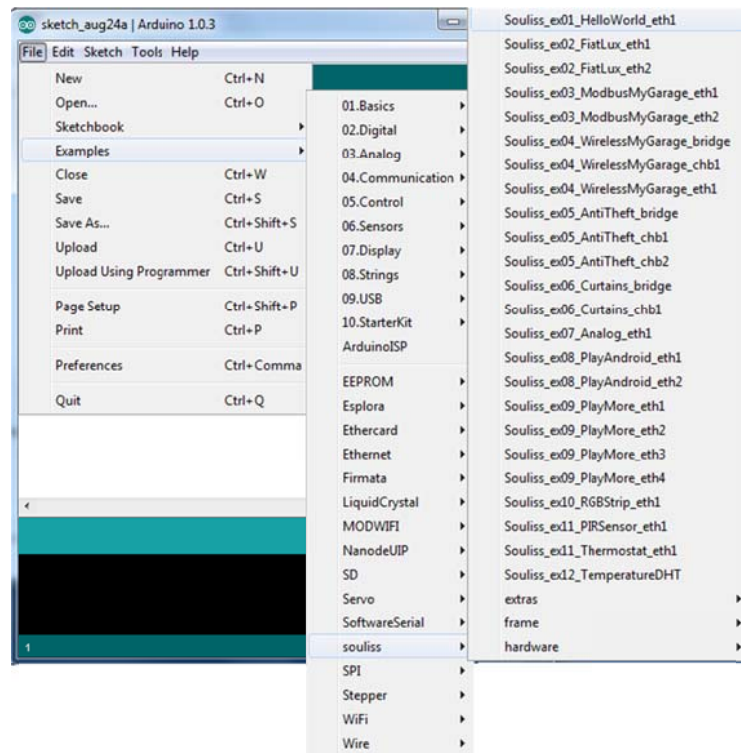
Descargue el último código disponible, obtendrá un archivo ZIP que contiene documentos y el código.



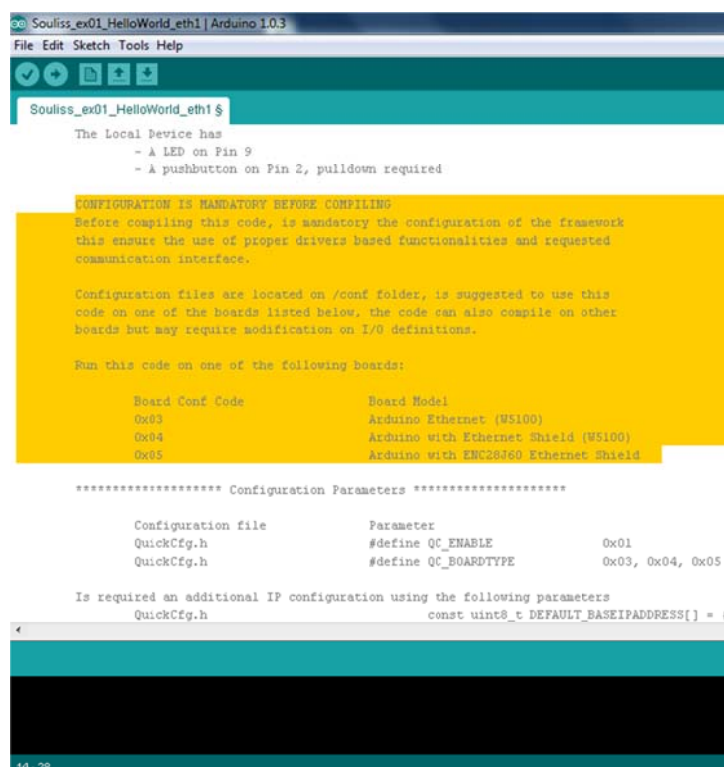
La carpeta /docs contiene algunos documentos en PDF, no son para principiantes, así que vaya directamente a la carpeta /souliss y cópiela y péguela en la carpeta libraries de su Arduino IDE.



Se sugiere evitar el uso del IDE Arduino en la carpeta Archivos de programa, utilice el acceso directo del escritorio en su lugar. Una vez copiada la carpeta /souliss en la carpeta /libraries del IDE, puede iniciar el IDE desde el acceso directo del escritorio.

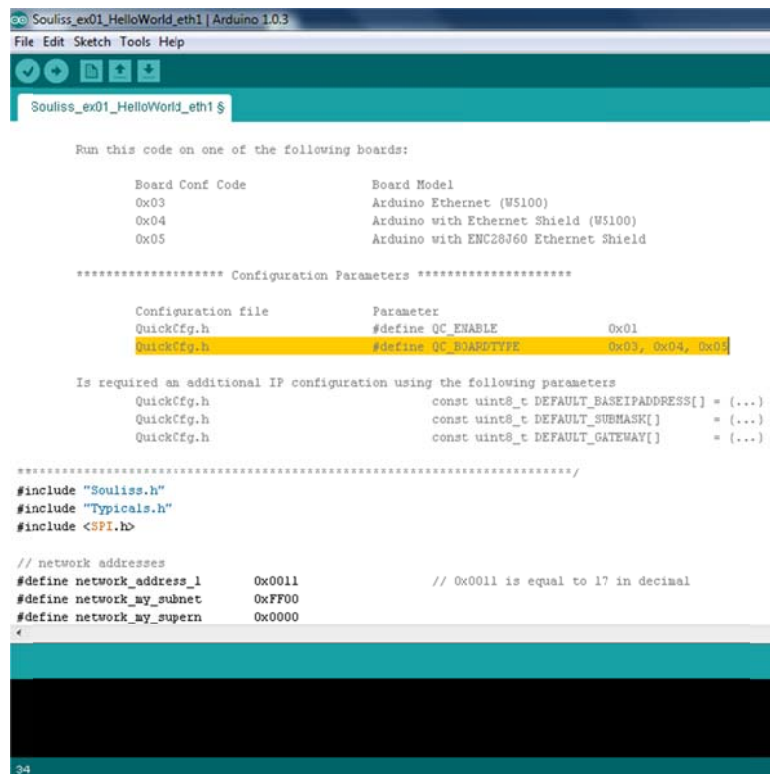


Ahora en Archivo → Ejemplos se encuentra el submenú souliss que contiene todos los ejemplos, comenzar primero con el Hello World en la parte superior de la lista.



Una vez abierto, en la parte superior del Sketch hay algunos comentarios (en gris) que describen el propósito del código y dan algo de información. Deben seguirse detenidamente antes de compilar y cargar el código.

Básicamente, Souliss ejecuta en varias plataformas y antes de compilación se debe de especificar cuáles son. Supongamos utilizar Arduino Ethernet (esta está en el listado de placas soportadas para este Sketch).



```
Souliss_ex01_HelloWorld_eth1 | Arduino 1.0.3
File Edit Sketch Tools Help

Souliss_ex01_HelloWorld_eth1 $

Run this code on one of the following boards:

Board Conf Code      Board Model
0x03                  Arduino Ethernet (W5100)
0x04                  Arduino with Ethernet Shield (W5100)
0x05                  Arduino with ENC28J60 Ethernet Shield

***** Configuration Parameters *****

Configuration file    Parameter
QuickCfg.h            #define QC_ENABLE      0x01
QuickCfg.h            #define QC_BOARDTYPE   0x03, 0x04, 0x05

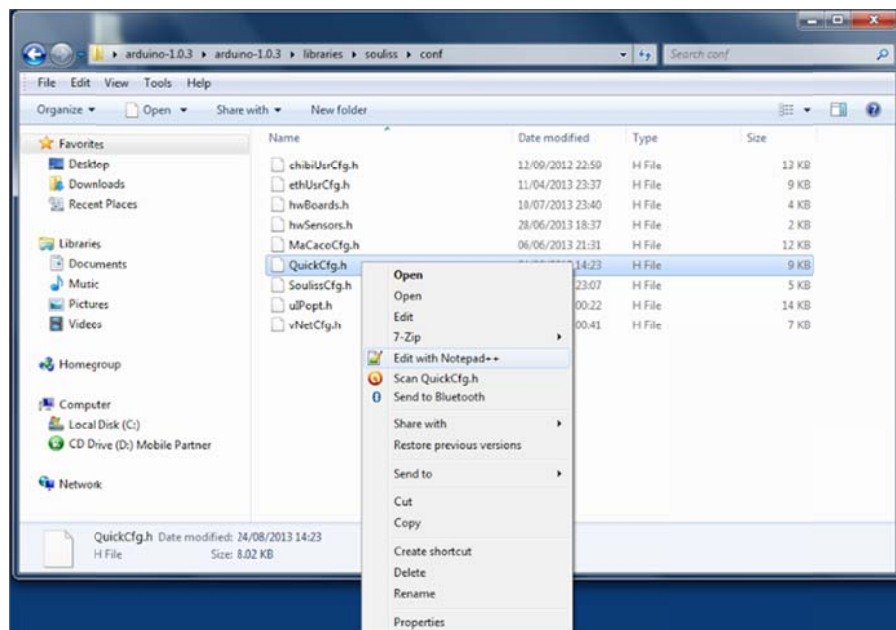
Is required an additional IP configuration using the following parameters
QuickCfg.h            const uint8_t DEFAULT_BASEIPADDRESS[] = (...)
QuickCfg.h            const uint8_t DEFAULT_SUBMASK[] = (...)
QuickCfg.h            const uint8_t DEFAULT_GATEWAY[] = (...)

*****/
#include "Souliss.h"
#include "Typicals.h"
#include <SPI.h>

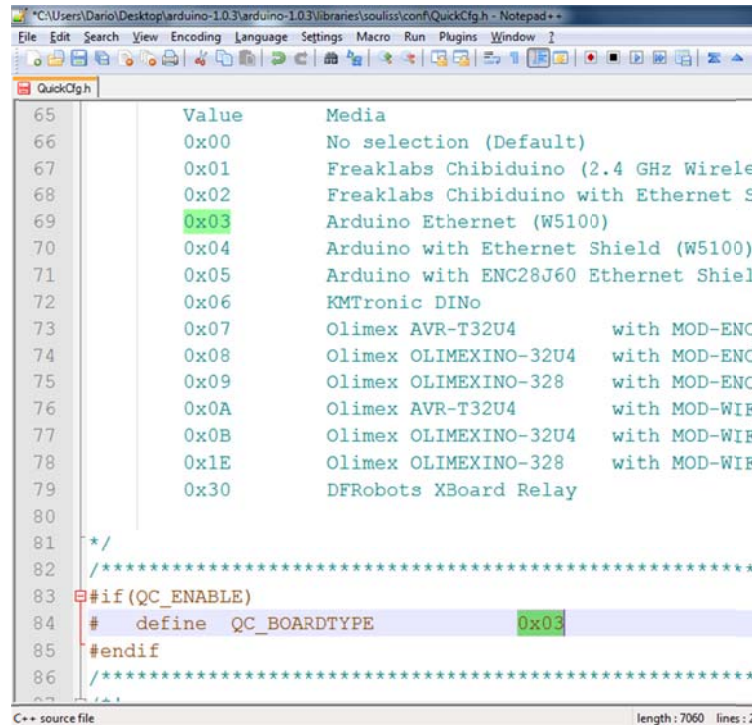
// network addresses
#define network_address_1 0x0011 // 0x0011 is equal to 17 in decimal
#define network_my_subnet 0xFF00
#define network_my_supern 0x0000

34
```

La línea resaltada dice que los parámetros de configuración se encuentran en el archivo QuickCfg.h, podemos seleccionar 0x03, 0x04 y 0x05, que como se ha visto antes, corresponde a la placa que vamos a utilizar. Arduino Ethernet tiene código 0x03.



Vamos a la carpeta /libraries del Arduino IDE y en /souliss/conf están todos los archivos de configuración disponibles. Durante la mayor parte del uso estándar, sólo se requiere [QuickCfg.h](#). Abrir con el Bloc de [Notepad++](#) para ver correctamente el código, no utilice el Bloc de notas de Windows.

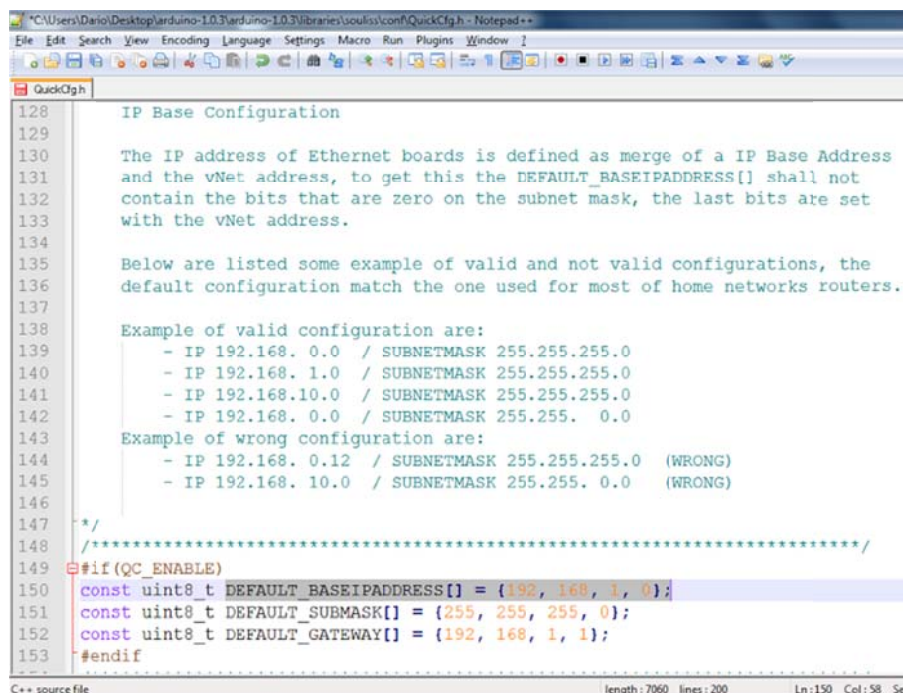


```
*CAUsers\Dario\Desktop\arduino-1.0.3\arduino-1.0.3\libraries\souliss\conf\QuickCfg.h - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

QuickCfg.h
65 Value Media
66 0x00 No selection (Default)
67 0x01 Freaklabs Chibiduino (2.4 GHz Wirele
68 0x02 Freaklabs Chibiduino with Ethernet S
69 0x03 Arduino Ethernet (W5100)
70 0x04 Arduino with Ethernet Shield (W5100)
71 0x05 Arduino with ENC28J60 Ethernet Shiel
72 0x06 KMTronic DINo
73 0x07 Olimex AVR-T32U4 with MOD-ENC
74 0x08 Olimex OLIMEXINO-32U4 with MOD-ENC
75 0x09 Olimex OLIMEXINO-328 with MOD-ENC
76 0x0A Olimex AVR-T32U4 with MOD-WIF
77 0x0B Olimex OLIMEXINO-32U4 with MOD-WIF
78 0x1E Olimex OLIMEXINO-328 with MOD-WIF
79 0x30 DFRobots XBoard Relay
80
81 */
82 /*****
83 #if (QC_ENABLE)
84 # define QC_BOARDTYPE 0x03
85 #endif
86 /*****
C++ source file length: 7060 lines: 2
```

Desplácese hacia abajo hasta el QC_BOARDTYPE y seleccione el código 0x03 Arduino Ethernet, guarde el archivo y vuelva a la Arduino IDE (no cierre el [Notepad++](#)).

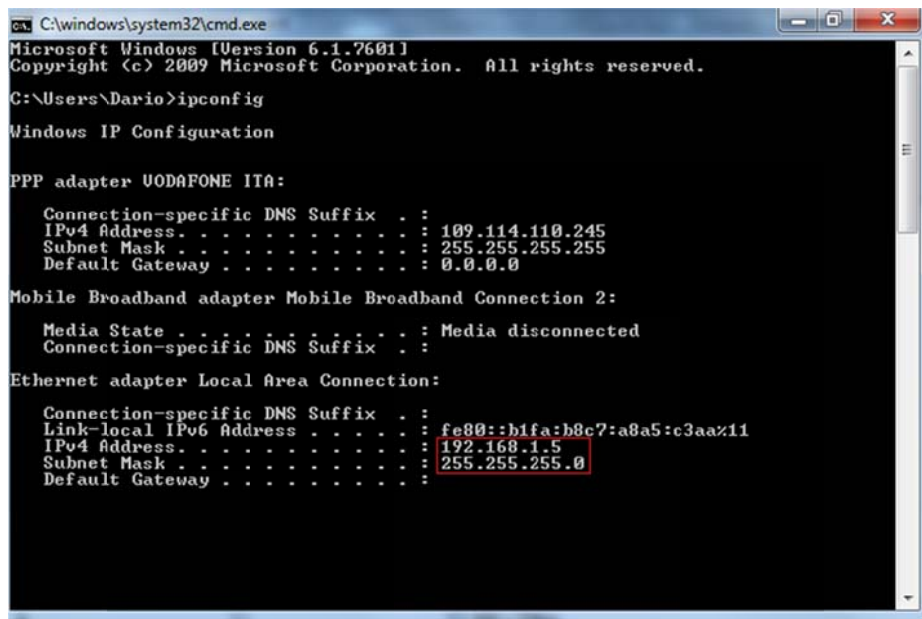
En el Sketch también se solicita que configure una direccion IP de base, que están disponibles en el archivo [QuickCfg.h](#).



```
*CAUsers\Dario\Desktop\arduino-1.0.3\arduino-1.0.3\libraries\souliss\conf\QuickCfg.h - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

QuickCfg.h
128 IP Base Configuration
129
130 The IP address of Ethernet boards is defined as merge of a IP Base Address
131 and the vNet address, to get this the DEFAULT_BASEIPADDRESS[] shall not
132 contain the bits that are zero on the subnet mask, the last bits are set
133 with the vNet address.
134
135 Below are listed some example of valid and not valid configurations, the
136 default configuration match the one used for most of home networks routers.
137
138 Example of valid configuration are:
139 - IP 192.168. 0.0 / SUBNETMASK 255.255.255.0
140 - IP 192.168. 1.0 / SUBNETMASK 255.255.255.0
141 - IP 192.168.10.0 / SUBNETMASK 255.255.255.0
142 - IP 192.168. 0.0 / SUBNETMASK 255.255. 0.0
143 Example of wrong configuration are:
144 - IP 192.168. 0.12 / SUBNETMASK 255.255.255.0 (WRONG)
145 - IP 192.168. 10.0 / SUBNETMASK 255.255. 0.0 (WRONG)
146
147 */
148 /*****
149 #if (QC_ENABLE)
150 const uint8_t DEFAULT_BASEIPADDRESS[] = {192, 168, 1, 0};
151 const uint8_t DEFAULT_SUBMASK[] = {255, 255, 255, 0};
152 const uint8_t DEFAULT_GATEWAY[] = {192, 168, 1, 1};
153 #endif
C++ source file length: 7060 lines: 200 Ln:150 Col:58 Sel
```


La descripción detallada de cómo configurar la dirección IP de base está disponible en el wiki, pero como regla general acaba de obtener su dirección local de PC y poner los últimos valores a cero.



```
C:\windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Dario>ipconfig

Windows IP Configuration

PPP adapter UODAFONE ITA:

    Connection-specific DNS Suffix  . : 
    IPv4 Address. . . . . : 109.114.110.245
    Subnet Mask . . . . . : 255.255.255.255
    Default Gateway . . . . . : 0.0.0.0

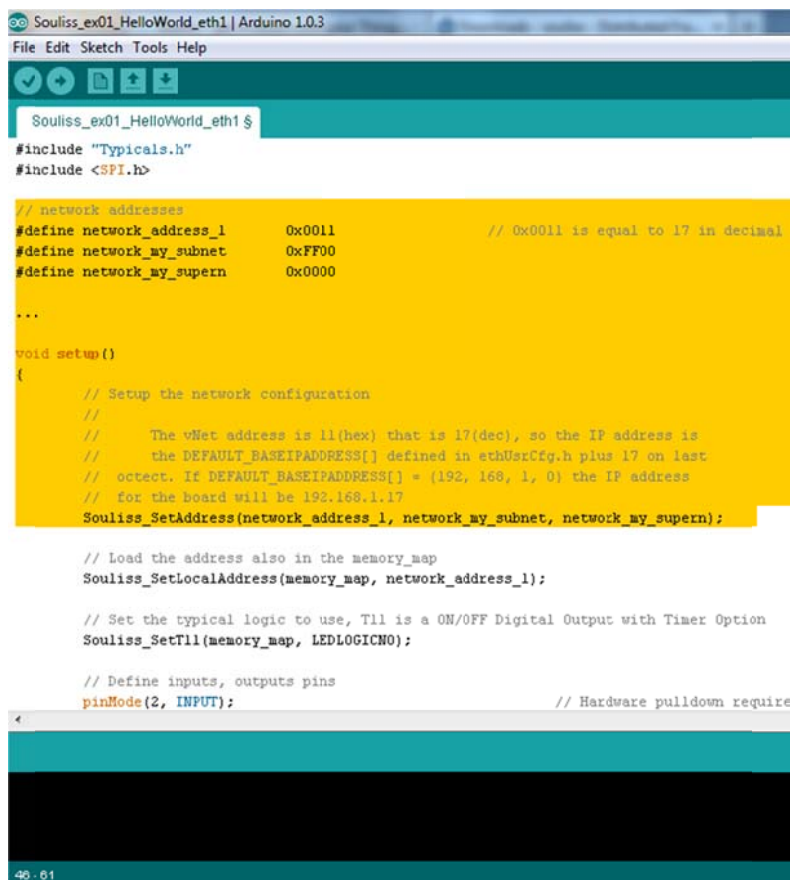
Mobile Broadband adapter Mobile Broadband Connection 2:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : 

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::b1fa:b8c7:a8a5:c3aa%11
    IPv4 Address. . . . . : 192.168.1.5
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . :
```

La configuración por defecto que figura en [QuickCfg.h](#) funciona para la mayoría de redes domésticas.



```
Souliss_ex01_HelloWorld_eth1 | Arduino 1.0.3
File Edit Sketch Tools Help

Souliss_ex01_HelloWorld_eth1 $

#include "Typicals.h"
#include <SPI.h>

// network addresses
#define network_address_1 0x0011 // 0x0011 is equal to 17 in decimal
#define network_my_subnet 0xFF00
#define network_my_supern 0x0000

...


void setup()
{
    // Setup the network configuration
    //
    // The vNet address is 11(hex) that is 17(dec), so the IP address is
    // the DEFAULT_BASEIPADDRESS[] defined in ethUserCfg.h plus 17 on last
    // octect. If DEFAULT_BASEIPADDRESS[] = {192, 168, 1, 0} the IP address
    // for the board will be 192.168.1.17
    Souliss_SetAddress(network_address_1, network_my_subnet, network_my_supern);

    // Load the address also in the memory_map
    Souliss_SetLocalAddress(memory_map, network_address_1);

    // Set the typical logic to use, T11 is a ON/OFF Digital Output with Timer Option
    Souliss_SetT11(memory_map, LEDLOGICN0);

    // Define inputs, outputs pins
    pinMode(2, INPUT); // Hardware pulldown require
}
```

Ahora tiene que entender cuál la dirección IP de su placa, en los Sketches se utiliza como dirección del nodo 0x0011 que en decimal es igual a 17, como regla general, la dirección IP de su placa es la dirección IP de base en el que el último número es el dirección de nodo. En este caso la IP de base es 192.168.1.0 y la dirección de nodo es 17, por lo que la dirección de su placa es 192.168.1.17.



```
//
// The vNet address is 11(hex) that is 17(dec), so the IP address is
// the DEFAULT_BASEIPADDRESS[] defined in ethUserCfg.h plus 17 on last
// octect. If DEFAULT_BASEIPADDRESS[] = {192, 168, 1, 0} the IP address
// for the board will be 192.168.1.17
Souliss_SetAddress(network_address_1, network_my_subnet, network_my_supern);

// Load the address also in the memory_map
Souliss_SetLocalAddress(memory_map, network_address_1);

// Set the typical logic to use. Ttl is a ON/OFF Digital Output with Timer Option
Souliss_SetTtl(memory_map, LEDLOGICNO);

// Define inputs, outputs pins
pinMode(2, INPUT); // Hardware pulldown required
pinMode(9, OUTPUT); // Power the LED
}

void loop()
{
    // The Souliss methods are scheduled in phases, this allow load
    // balance and proper timing.

    if(abs(millis()-tar_fast) > time_base_fast)
    {
        tar_fast = millis();
        phase_fast = (phase_fast + 1) % num_phases;
    }
}
```

Debe asegurarse de que ningún otro dispositivo tiene esta dirección IP, si es así, cambie la dirección del nodo como 0x00nn donde “nn” es un número elegido en hexadecimal.

A continuación, guarde el archivo [QuickCfg.h](#) y cargue el Sketch desde el Arduino IDE a su placa.



```
C:\windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Dario>ping 192.168.1.17

Pinging 192.168.1.17 with 32 bytes of data:
Reply from 192.168.1.17: bytes=32 time=13ms TTL=128
Reply from 192.168.1.17: bytes=32 time=1ms TTL=128
Reply from 192.168.1.17: bytes=32 time<1ms TTL=128
Reply from 192.168.1.17: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.17:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 13ms, Average = 3ms

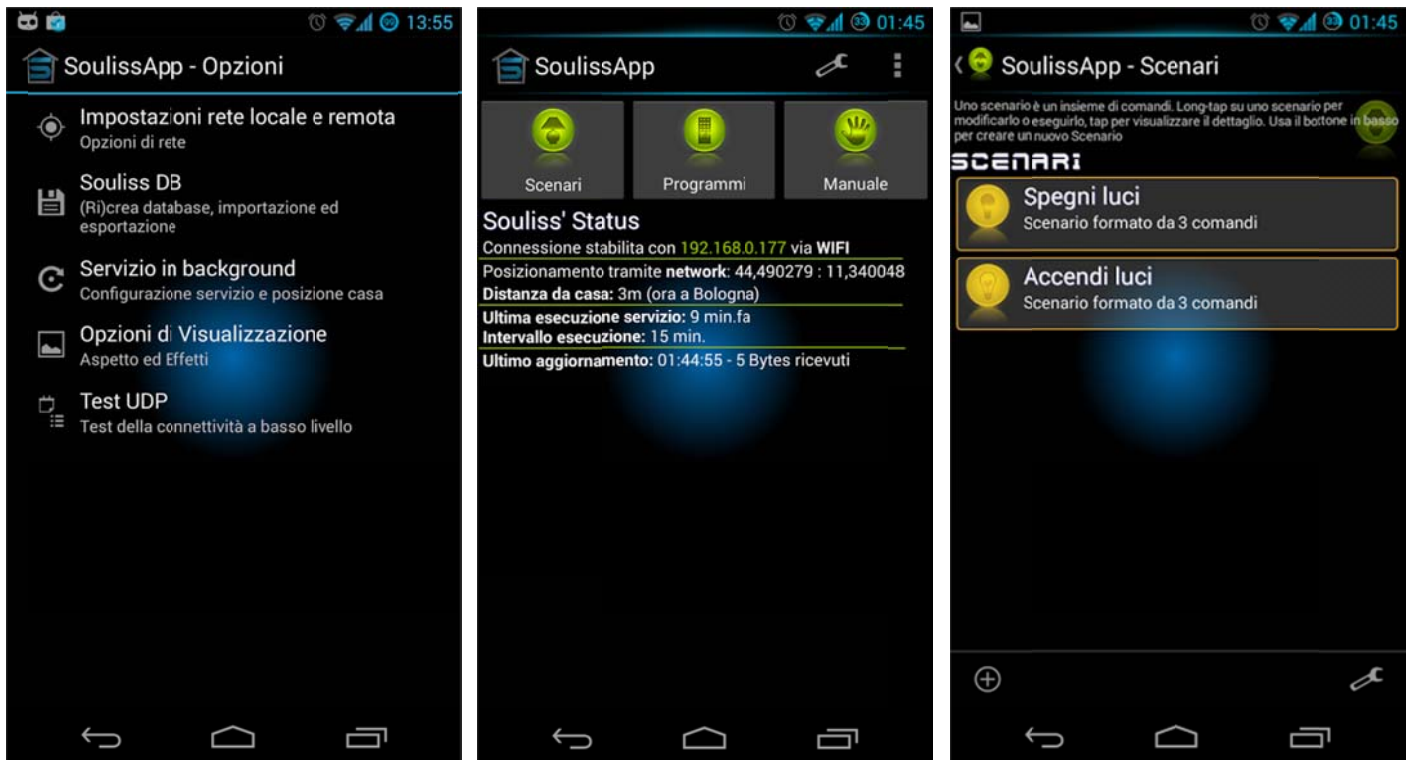
C:\Users\Dario>
```

Si todo va bien, puede hacer ping a su placa. Su primer nodo Souliss está online, simplemente recuerde esta dirección IP y pásala a la [SoulissApp](#).

APLICACIÓN DE SOULISS PARA ANDROID

INSTALACION

La aplicación de Souliss para Android se encuentra disponible en la página oficial de Google Play Market, ocupa alrededor de 4 MB de espacio en la sdcard. La aplicación es compatible con Android > 2.3 (Gingerbread) y permite enviar casi todos los comandos de Souliss con facilidad. La aplicación se descargará los datos de los nodos de Souliss y auto-configurarara su estructura.



OPCIONES Y CONFIGURACION

El proceso de configuración es muy sencillo, en un par de toques podrás controlar nodos Souliss con el smartphone o la tableta.

Una vez instalado, use el menú de opciones configurar la dirección IP de su nodo Souliss Gateway. La dirección privada es obligatoria, ya que SoulissApp tiene que saber que ocurre incluso cuando se utiliza la conexión no local (ej. 3G). La aplicación comprobará la conectividad y luego ingresar los datos en una base de datos local que contiene toda la [estructura de datos](#) en tu red. Para ello, sólo tiene que ir a configuración → Base de Datos Souliss y toque la función correspondiente; la base de datos local será producida y usted estará listo para usar la App.

Opcionalmente, si estás en casa, establezca la ubicación de inicio y activar la ejecución en segundo plano (desactivado por defecto). Esto permitirá a la aplicación mantenerse sincronizada.

ESCENAS

La aplicación incluye escenas, se pueden crear y modificar. Una escena es un conjunto de comandos que se emiten en secuencia. Por ejemplo, puede definir una lista de comandos a ejecutar cuando se vaya a la cama, llamar a esta escena "nocturna", y ejecutarlo con un solo toque.

Los comandos pueden o bien ser masivos o individuales: los primeros que se ejecutan en todos los dispositivos del mismo tipo (es decir, todas las luces), mientras que los comandos individuales sólo implican un dispositivo concreto (por ejemplo, las luces del dormitorio).

Las escenas pueden ser renombradas y un icono puede estar asociados a la misma, con el fin de ayudar al usuario a identificarla y recordarla.

PROGRAMAS

Los programas también se hacen por comandos, pero se desencadenan en tres diferentes tipos de eventos:

- Programaciones horarias: se ejecutan en un momento especificado del día, y pueden repetirse a intervalos regulares.
- Programas posicionales: detectar su posición, y se ejecutan cuando se vaya o cuando vuelves a casa.
- Programas basados en sensores: se ejecutan cuando se alcanza un cierto valor definido por el usuario. Por ejemplo, usted puede decidir para encender el aire acondicionado si la temperatura es demasiado alta.

Los programas se deben realizar aun cuando Souliss se está ejecutando en segundo plano, por lo que una notificación informará al usuario cuando algún comando se ejecute.

MODO MANUAL

El modo manual le permite manipular los dispositivos Souliss directamente, informando sobre el estado los dispositivos actuales. Aparecerá en la lista de nodos Souliss, puede pulsar sobre uno de ellos para ver los detalles de nodo y obtener el control.

Para cada dispositivo se muestran sus controles y/o una pantalla de detalle adicional, por ejemplo un gráfico (para sensores) o un control preciso para dispositivos particulares (luces RGB, aparatos de aire acondicionado). En la pantalla de detalles del nodo, es posible cambiar el nombre de nodo y seleccionar el icono del nodo, así como los nombres y los iconos de dispositivos.

CARACTERISTICAS

- Interfaz Souliss para Android
- Detecta automáticamente los nodos Souliss y los dispositivos
- Definir Escenarios y Programas (programaciones, geo-referenciados, basados en sensores)
- Servicio en segundo plano
- Personalizar objetos, renombrar nodos, dispositivos e iconos