El APK de Android • Juegos SEGA en tu ODROID • Juegos Linux

# ODROID Magazine

Año Uno
Num. #9
Sep 2014

# CONSTRUYE TU PROPIO WALL-E

## EL ADORABLE ROBOT DE PIXAR VUELVE A LA VIDA CON ODROID-U3

- FUNDAMENTOS DE BASH
- FREEDOMOTIC
- PARTE METEOROLOGICO
- CLUSTER U3 DE 10-NODOS
- ODROID-SHOW

## IMPRIME EN 3D UN SISTEMA DE JARDINERIA BASADO EN ODROID


## Y ADEMAS UN ORDENADOR PORTATIL FUTURISTA

# Qué defendemos.

Nos esmeramos en presentar una tecnología punta, futura, joven, técnica y para la sociedad de hoy.

Nuestra filosofía se basa en los desarrolladores. Continuamente nos esforzamos por mantener estrechas relaciones con éstos en todo el mundo.

Por eso, siempre podrás confiar en la calidad y experiencia que representa la marca distintiva de nuestros productos.

Simple, moderno y único.
De modo que  tienes a tu alcance lo mejor

HARDKERNEL

---

Pollin ELECTRONIC

Ahora estamos enviando los dispositivos ODROID U3  a los países de la UE! Ven y visita nuestra tienda online!

Dirección: Max-Pollin-Straße 1
85104 Pförring Alemania

Teléfono & Fax
telf : +49 (0) 8403 / 920-920
email : service@pollin.de

Nuestros productos ODROID se pueden encontrar en:
http://bit.ly/1tXPXwe

# EDITORIAL

Con la introducción de ODROID-W y la placa meteorológica ODROID, se han llevado a cabo varios proyectos publicados recientemente en los foros ODROID que implican domótica, iluminación ambiental y robótica. Este mes, presentamos varios de estos proyectos, como son la posibilidad de poder ir a pescar el fin de semana, la construcción de una caja a medida para un portátil, el cuidado del jardín de forma remota y la construcción de una fiel reproducción del robot favorito de todo del mundo, ¡Wall-E!

Hardkernel hará una demostración del nuevo XU3 en ARM Techcon del 1 al 3 de octubre de 2014 en San José, California. Visite el stand si desea conversar con algunos de los miembros del equipo Hardkernel y de ODROID Magazine. El coste de la entrada para la exposición es actualmente de 59$ disponible en www.armtechcon.com

La reciente publicación del XU3 octa-core ya cuenta con varios sistemas operativos modernos disponibles, incluyendo Android y Ubuntu. El grupo de ArchLinux ya ha publicado las instrucciones para la elaboración de ArchLinux para ARM (ALARM) y para el XU3 en http://bit.ly/1tS2xNs. Hardkernel ofrece Android 4.4 para su descarga en http://bit.ly/1qMA6Oq, El uruario @voodik de los foros ODROID publicó CyanogenMod 11 en http://bit.ly/1qMA6Oq y Ubuntu 14.04 está disponible en http://bit.ly/ 1s06GZW.

Si todavía no has encargado uno, la XU3 es el ordenador más rápido que Hardkernel ha hecho nunca, ya que es capaz de usar los ocho núcleos al mismo tiempo, mejorando el diseño del clúster del XU original alternando entre la alta eficiencia de los núcleos A7 y el rendimiento de los núcleos A15. También es compatible con USB 3.0 y eMMC 5.0, y ofrece lo última GPU Mali T-628 MP6 con OpenGLES 3.0 y OpenCL 1.1. Está disponible en la tienda Hardkernel en http://bit.ly/YGEnc2.

ODROID Magazine, que se publica mensualmente en http://magazine.odroid.com/, es la fuente de todas las cosas ODROIDianas. • Hard Kernel, Ltd. • 704 Anyang K-Center, Gwanyang, Dongan, Anyang, Gyeonggi, South Korea, 431-815 • fabricantes de la familia ODROID de placas de desarrollo quad-core y la primera arquitectura ARM “big.LITTLE” del mundo basada en una única placa.
Únete a la comunidad ODROID con miembros en más de 135 países en http://forum.odroid.com/ y explora las nuevas tecnologías que te ofrece Hardkernel en http://www.hardkernel.com/.

HARDKERNEL

# ODROID Magazine

## Rob Roy, Editor Jefe

Soy un programador informático que vive y trabaja en San Francisco, CA, en el diseño y desarrollo de aplicaciones web para clients locales sobre mi cluster de ODROID. Mis principales lenguajes son jQuery, angular JS y HTML5/CSS3. También desarrollo sistemas operativos precompilados, Kernels persona-lizados y aplicaciones optimizadas para la plataforma ODROID basadas en las versiones oficiales de Hardkernel, por los cuales he ganado varios Premios. Utilizo mi ODROIDs para diversos fines, como centro multimedia, servidor web, desarrollo de aplicaciones, estación de trabajo y como plataforma de juegos. Puedes echar un vistazo a mi colección de 100 GB de software e imágenes ODROID en `http://bit.ly/1fsaXQs`.

## Bo Lechnowsky, Editor

Soy el presidente de Respectech, Inc., Consultoría tecnológica en Ukiah, CA, EE.UU. que fundé en 2001. Con mi experiencia en electrónica y programación dirijo a un equipo de expertos, además de desarrollar soluciones personalizadas a empresas, desde pequeños negocios a compañías internacionales. Los ODROIDs son una de las herramientas de las que dispongo para hacer frente a estos proyectos. Mis lenguajes favoritos son Rebol y Red, ambos se ejecutan en los sistemas ARM como el ODROID-U3. En cuanto a aficiones, si necesitas alguna, yo estaría encantado de ofrecerte alguna de la mías ya que tengo demasiadas. Eso ayudaría a que tuviese más tiempo para estar con mi maravillosa esposa de y mis cuatro hijos estupendos.

## Bruno Doiche, Editor Artístico

Consiguió sus habilidades informáticas después de lograr que una fibra óptica volviera a la vida, lograr que su Macintosh volviese de la muerte, lograr que una PS3 volviese de la muerte, lograr que el T400 de su novia volviese de la muerte (una transferencia de datos dd al viejo estilo), y liando con las entrañas de su permanente centro de datos de trabajo.

## Nicole Scott, Editor Artístico

Nicole es una experta en Producción Transmedia y Estrategia Digital especializa en la optimización online y estrategias de marketing, administración de medios sociales y coordinación de equipo, así como la producción multimedia impresa, TV, cine y web. Nicole es experta en diseño gráfico y web, gestión de redes sociales y publicidad, edición de vídeo y maquetación DVD. Dispone de un ODROID U3 que usa para aprender Linux. Ella vive en el área de la Bahía de California, y disfruta haciendo senderismo, acampada y tocando música. Visite su web en **http://www.nicolecscott.com**

## Manuel Adamuz, Editor Español

Tengo 31 años y vivo en Sevilla, España, y nací en Granada. Estoy casado con una mujer maravillosa y tengo un hijo. Hace unos años trabajé como técnico informático y programador, pero mi trabajo actual está relacionado con la gestión de calidad y la ecnología de la información: ISO 9001, ISO 27001, ISO 20000 Soy un apasionado de la informática, especialmente de los microordenadores como el ODROID, Raspberry Pi, etc. Me encanta experimentar con estos equipos y traducir ODROID Magazine. Mi esposa dice que estoy loco porque sólo pienso en ODROID . Mi otra gran afición es la bicicleta de montaña, y de vez en cuando participo en competiciones semiprofesionales.

# INDICE

# EJECUTAR JUEGOS DE SEGA EN HD 1080P

## UN VIAJE AL PASADO

**Por Jeremy "Cartridge" Kenney**

Un ODROID es todo lo que necesitas para ejecutar de nuevo tus juegos favoritos de Sega. Opcionalmente necesitarás un adaptador de mandos de Sega Genesis para que tu mando pueda ser reconocido por ODROID y por supuesto, conectarlo al puerto USB. Luego, puedes hacerte con una copia de DGEN-SDL para ODROID, que está compilado y disponible en los foros HardKernel, `http://bit.ly/1pgoyO8` y empeza a disfrutar de los juegos de Sega.

Las funciones únicas de DGEN la convierten en una extraordinaria aplicación que incluye una amplia variedad de opciones para personalizar la emulación. DGEN también puede hacer uso de códigos GameGenie, así que si todavía conservas un manual de GameGenie o tienes escritos los códigos en la sección de "Notas" de tu manual de juego, puedes usarlos con DGEN.

Los modos de pantalla completa de NTSC y PAL funcionan perfectamente, tanto para América del Norte y Europa como para NTSC Japón, para cualquier TV que disponga de salida de vídeo. OpenGL esta implementado, pero no compilado, porque los ordenadores ODROID hacen uso de OpenGL-ES que no se puede trasladar tan fácilmente a OpenGL desde que la versión ES es un subgrupo de OpenGL.

Permite hacer capturas de pantalla y grabar vídeo. Sin embargo la grabación no se realiza en formato AVI o MPG, sino que se hace en un formato propietario. Esto significa que no podemos reproducir el vídeo en un reproductor corriente, sólo a través de DGEN. Funciona bastante bien permitiendo avanzar, retroceder e incluso pausar la reproducción.

Utiliza el sonido de 16 bits para mantener los gráficos y sonido originales de Sega, pero puedes modificar el muestreo desde los 8000Hz a 48000Hz para conseguir una mejor calidad. También puedes cambiar la salida del color en los modos de 8, 15, 16, 24 y 32 bpp dándote una gran variedad de bits donde poder elegir. Soporta archivos comprimidos, incluidos rar y zip.

Scale2x y HQX solo soportan resoluciones de escritorio y para tu comodidad está compilado para que funcione en resoluciones CRT. Por último, el depurador M68K esta implementado para desarrolladores con el fin de depurar juegos y aplicaciones que se ejecuten con el chip 68K.

He incluido un archivo de texto en el mensaje original del foro para asociar los botones y adaptarlos a tu mando en particular. DGEN ya debería detectar el mando y auto-asignar los botones. Para los mandos no compatibles, consulta el archivo de texto de asignación de botones. Aunque está incompleto, estamos reuniendo todos los mandos posibles para determinar si son detectados correctamente. La mayoría de los mandos, gamepads y joysticks del 2.003 son identificados correctamente sin necesidad de una configuración adicional, incluyendo el mando Sega Genesis mencionado anteriormente.

Para instalar DGEN, descomprime el paquete y escriba lo siguiente en el terminal, reemplazando el archivo .deb por el nombre del paquete:

```
$ sudo dpkg -i Nombredetupaquete
debian.deb
```

Si dispones de un mando, lo mejor es conectarlo antes de ejecutar la aplicación.

**Sega Genesis, ¡La consola que nos presento Sonic!**


Juego de Sega en ejecución

Para iniciar DGEN, escriba lo siguiente, sustituyendo el nombre de archivo por la ROM seleccionada:

```
dgen -f nombrerom.bin
```

## Optiones

Usa la opción -f para activar la pantalla completa y -G para activar una resolución específica, por ejemplo:

```
dgen -G 1279x719
```

Esto generará una resolución de 1024x768 en una ventana de escritorio. HQX y Scale2x pueden activarse pulsando F6 y F5 para los filtros de "TV malas", dando la posibilidad de disponer de una salida de vídeo de baja resolución, ¡como en los viejos tiempos!

La opción -R te ayudará a ejecutar el juego en la región que elijas. Los posibles códigos son "E" para Europa, "U" para USA/Canadá y "J" para Japón:

```
$ dgen -f -R E nombrerom.bin
```

La opción -D reproducirá tu vídeo grabado:

```
$ dgen -f -d demoname romname.bin
```

Ahora tu ODROID está listo para ejecutar juegos de Sega Genesis en tu TV HD, viejo TV CRT o en cualquier otro monitor. Si no tienes un viejo televisor para jugar o tu TV HD tiene demasiada definición, activa los filtros CrapTV para que parezca más auténtico. ¡Diviértete jugando a Sega en tu ODROID!

# JUEGOS LINUX

## EJECUTAR JUEGOS NATIVOS EN ODROID PARTE I

**Por Tobias Schaaf**

En anteriores artículos, presente una visión general de los diferentes emuladores disponibles en la imagen GameStation Turbo, que soportan miles de juegos emulando diferentes sistemas de consola como la SNES o PS1. Por otro lado y puesto que GameStation Turbo utiliza Linux como sistema operativo de fondo, me gustaría analizar más detenidamente algunos de los juegos que existen para Linux y que se ejecutan de forma nativa, sin un emulador. Todas las fotos que incluye este artículo fueron tomadas con un ODROID.


### DOOM 3 - Shooter en primera persona de ciencia ficción y terror (18+)

Para los que todavía no lo saben, Existe un impresionante juego de disparos en primera persona disponible para ODROID. @AreaScout exporto DOOM 3 a la plataforma ODROID y fue capaz de hacerlo funcionar de forma nativa con OpenGL ES. En Doom 3, juegas con un soldado que recientemente llegó a la base ARS, un centro de investigación enorme en el que muchos científicos trabajan en diferentes proyectos, incluyendo el teletransporte. Usando esta técnica los científicos descubren una nueva dimensión, pero ¡Algo sale mal! y los demonios empiezan aparecer y matan a los seres humanos o los transforman en monstruos.

¡Se arma la de Dios! Tienes que luchar y abrirte paso a lo largo de todos los niveles. El juego viene con una gran cantidad de secuencias que se supone que asustan e intimidan al jugador con efectos impactantes. Junto con el entorno oscuro del juego, posiblemente te asustarás de vez en cuando.

@AreaScout implementó un nuevo shader en el juego para corregir un problema con el gamma y el brillo. Con este ajuste, puedes iluminar el entorno del juego, facilitando el poder encontrar objetos y ver los rincones donde el enemigo podría estar al acecho, listo para asustarte y atacarte. Doom 3 es en definitiva el videojuego de disparos en primera persona y de terror que falta en tu colección.

### Homeworld - Estrategia espacial en tiempo real

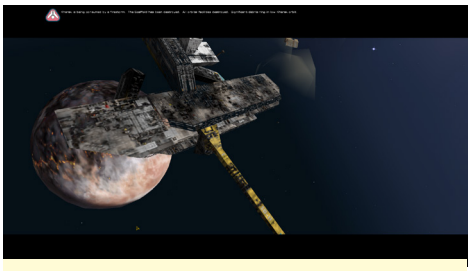
Otro juego que se puede ejecutar de forma nativa en ODROID es un asombroso juego de estrategia en tiempo real (RTS) llamado Homeworld. En el cual formas parte de un grupo de personas que descubrió los planos de una vieja nave durante una expedición, además

**Recolector de recursos y nave nodriza (Homeworld)**

de un Hiperespacio que permite viajar a través del espacio. Descubren el origen de su propia especie en un mundo llamado HIIGARA (el hogar), y deciden construir una gran nave espacial llamada la " mothership " para viajar a ese mundo.

Sin embargo, poco después de terminar la mothership y realizar su primer salto al hiperespacio, algo sale mal. La nave con la que se suponía que debías reunirte ha sido destruida por piratas y cuando vuelves a tu planeta, te encuentran que ha sido devastado junto con la estación espacial. Así que, tienes que construir una flota para defenderte, vengar a tu gente y encontrar el camino a HIIGARA. Pronto encontrarás nuevos aliados, enemigos y otras especies.

Homeworld es un juego muy bueno, con magníficos gráficos y una gran banda sonora que incluye la famosa Adagio for Strings de Samuel Barber con el famoso Coro de Niños de Viena. El juego incorpora muchos detalles en los diferentes modelos de naves espaciales asi como en los efectos, fondos y planetas.



**Astillero y planeta destruidos (Homeworld)**

**Empezando a construir una flota para defenderse (Homeworld)**

**Vista detallada en la que se dispara a una nave enemiga (Homeworld)**

La historia de Homeworld es muy emocionante ya que tienes que investigar nuevas tecnologías y naves, planificar tus movimientos, gestionar los hogares con los escasos recursos de los que dispones y proteger tu flota al tiempo que avanzas a través de las líneas enemigas. Si te gustan los juegos de estrategia en tiempo real y el género espacial, este es un juego imprescindible que te mantendrá ocupado durante horas y horas.


## EDuke 32 - Shooter en primera persona

**"¡Ven por más!"** -- Duke Nukem

Duke Nukem es un personaje muy famoso en la historia de los juego. Es el típico tipo duro que salva al mundo y a todas las chicas, de una invasión alienígena malvada. Siempre será recordado por frases como " ¿A qué esperas? ¿A las navidades? " Con el recientemente exportado EDuke32, Duke Nukem vuelve a la vida y esta vez en alta definición con modelos poligonales reales en 3D. Cuando el juego salió a la luz para PC, jugué a la versión para DOS. Todavía me pregunto cómo fui capaz de jugar al juego sin un ratón, mirando hacia arriba y hacia abajo presionando las teclas del teclado mientras me movía a la izquierda y a la derecha, a la vez que trataba de evitar los disparos del enemigo.

Originalmente, el juego ofrecía un mundo en 3D con sprites en 2D como personajes y objetos, lo cual era bastante impresionante por aquel entonces. Todavía se ve bastante bien con el filtrado trilineal.

**Duke Nukem en frente de un espejo "¡Maldición ... Parezco muy bueno!"**

Como curiosidad, los algoritmos de los espejos no eran tan fáciles de ejecutar en DOS, así que los programadores crearon otro mundo idéntico y duplicaron todos los personajes y movimientos. Cuando mirabas a un espejo en realidad no veías un reflejo, sino otro Duke Nukem copiando tus movimientos.

La captura de pantalla muestra cómo se relleno por aquel entonces el mundo virtual con sprites, ya que las tarjetas 3D no eran muy comunes y todo lo hacia la CPU. Afortunadamente, esos días han terminado y ahora tenemos una nueva versión de Duke Nukem en 3D con texturas en alta resolución. Si no has jugado al Duke Nukem 3D antes, la nueva versión merece la pena, ofrece un montón de acción, una buena jugabilidad y un poco de literatura erótica de aquí y allá.

No te olvides de las famosas frases del propio Duke Nukem que lo convirtieron en el personaje de juegos más cabrón de todos los tiempos. Pero recuerda, el juego fue creado para un público adulto. Ahora, sólo queda una cosa por hacer: "¡Viva el rey, baby!"

# CONSTRUIR UN PORTATIL TODO EN UNO

## LLEVATE TU U3 A CUALQUIER PARTE

**Por Daniele S.**

**La caja del portátil terminada, con altavoces instalados en la parte inferior del frontal del plexiglás transparente, un útil I/O shield para el desarrollo de prototipos de hardware y un bonito monitor de 10.1" HD**

El desarrollo de mi propio ordenador portátil ODROID comenzó cuando se me rompió mi antiguo portátil de un único núcleo. Busqué por Internet con el fin de reemplazarlo y descubrí el fantástico ODROID U3, que es lo suficientemente potente como para ser utilizado en lugar de un ordenador normal. Decidí construir un portátil con un ODROID-U3 y hacer una caja de plexiglás a medida.

Para la pantalla, compre un buen monitor de 10.1 pulgadas, así como un par de accesorios como el I/O shield y el adaptador wifi, que es todo lo que necesitaba para empezar a construir mi portátil "todo-en-uno". Por otro lado, tenía muchos periféricos sin utilizar para adaptárselos. Aunque había visto varios PCs hechos con madera, opté por utilizar plexiglás que es muy fácil de trabajar.

En primer lugar, use lápiz y papel para esbozar el proyecto. Mi diseño no era nada complicado, así que medí y corté directamente sobre el plexiglás. La trasparencia del plexiglás me facilito las mediciones. Por ejemplo, para tomar la medida de los agujeros del ODROID, simplemente puse la placa sobre la hoja de plexiglás y una vez invertida, marque los cuatro puntos dónde taladrar.

Para el plexiglás, recomiendo usar un grosor de al menos 5 mm con el fin de reforzar bien el fondo y la estructura. Para los paneles superiores es suficiente con 4mm. Contar con la maquinaria de bricolaje todavía resulta algo caro (a menos que te la preste un vecino), así que utilice muchas herramientas de las que solemos disponer en casa o en el garaje.

* **Sierra caladora**
* **Destornillador**
* **Secador de pelo**
* **Lima de uñas**
* **Cinta métrica**
* **Marcador**

Para aquellos que no están familiarizados con el plexiglás, tiende a fundirse a altas temperaturas. La primera vez que hice un corte en plexiglás, fui demasiado rápido y pude ver que tras la hoja de corte el material se derretía y se enfriaba casi al mismo tiempo. Para cuando finalice el corte ya se había recompuesto como si no hubiese sido cortado nunca.

He aprendido a mantener baja la velocidad de la hoja de corte, encendiéndola y apagándola continuamente para que se moviera lentamente. Para los cortes rectos, use una regla de cierta longitud y coloque una hoja más ancha para que me ayudara a mantener el ángulo recto. Usar una hoja más estrecha me permitía crear los círculos para los cables, conectores e interruptores.

Es importante tener a mano un buen vaso de agua - no para beber, sino para la hoja de corte. Cada 2-3 cm de corte, quitaba la hoja y la dejaba enfriar en el vaso de agua que además la lubrica. Cuando taladres procura mantener todo limpio y mojado, y coloca un pedazo de madera debajo del plexiglás. La madera evita que el plexiglás se rompa cuando el taladro perfora la parte inferior.

Luego pasé algún tiempo curvando los bordes de la caja, usando un secador de pelo y aplicando aire caliente sobre el plexiglás. Transcurrido un tiempo se ablanda pudiendo aplicar presión (se necesita muy poca). En el lado del panel donde empecé a doblar aplique aire de nuevo lentamente. Procura ser delicado a la hora de hacer esto: Si el plexiglás está demasiado frío, puede agrietarse o derretirse si está demasiado caliente.

Una vez que la pieza se inclina hacia la posición deseada, utilice un ventila-

dor para mantener el plexiglás con esa forma. También me ayudó usar paneles de madera anclados con abrazaderas para conseguir una línea recta a seguir en el pliegue. Recomiendo practicar en trozos de plexiglás desechables. Una vez que le cojas el truco, resulta muy entretenido construir de esta forma una caja a medida para un portátil.

Para pegar los paneles entre sí, he usado algunos tornillos autoperforables de un viejo PC. Tras comprobar que todo encajaba bien, los desmonté y pasé a la pintura. Aplique una capa al interior de la parte trasera del panel, dando un efecto espejo. Si escribes en el panel de plexiglás con cinta, retirarla una vez que la pintura se haya secado para obtener así un bonito efecto de transparencia y de iluminación por la noche.

Puesto que instalé la placa con el chip hacia abajo, coloque dos piezas de plástico cerca de los dos LEDs para reflejar la luz. No es la solución más práctica del mundo, pero es simple y funcional. Por último, añadí dos altavoces procedente de mi viejo ordenador portátil para la reproducción de música.

Aunque en las fotos parece que el portátil está terminado, el proyecto no se ha completado todavía, necesito instalar los botones de encendido y apagado en la parte frontal con el fin de tener una única fuente de alimentación externa de 12V. También tengo la intención de añadir un pack de baterías de litio, para convertirlo en un ordenador portátil en si mismo.

He trabajado en el proyecto durante mucho tiempo, dando prioridad a la funcionalidad más que a la estética. No obstante, en cuestión de estética todo es mejorable, así que coge las herramientas de tienes en casa y monta tu propia caja a medida para tu ODROID.

# FUNDAMENTOS DE BASH

## SHEBANGS Y SHEBANGS

**Por Tynan Overstreet**


**Las habilidades básicas de bash son el punto de partida para convertirse en un gurú de Linux**

Bash es una herramienta muy útil para automatizar tareas administrativas en tu ODROID. Un script puede ser algo rápido para, simplemente terminar un trabajo o ejecutarse cada vez que arranque tu ordenador.

### Desarrollo de un script bash

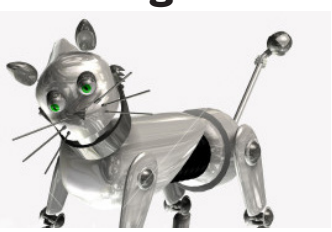
Todo script bash debe empezar con:

```
#!/bin/bash
```

Esto le permite al sistema conocer qué intérprete usar. Curiosamente, el simbolo "#!" se conoce como un "shebang". Sugiero utilizar la palabra tanto como te sea posible, ya que es usada muy poco, aparte de en las viejas canciones de Ricky Martin. El coro de su canción hacía referencia continuamente a los Scripts bash. En este artículo aprenderás a usar bash para:

- Hacer algo cada vez que tus ODROID se inicien
- Hacer cálculos y concatenar cadenas
- Aceptar argumentos
- Ejecutar un bucle
- Lanzar un grupo de subprocesos con diferentes argumentos

### Construir un gato robot



Para el primer ejemplo, vamos a hacer un script combinando dos cadenas.

**1. Crea un archivo llamado meow.sh**

```
#!/bin/bash

uno="me" # i'm a comment… uno
references a string
dos="ow"  # you probably get
where I'm going with this
echo "${uno}${dos}" # variables
are referenced ${…}
```

**2. Tras crear el archivo, escriba lo siguiente en la línea de comandos:**

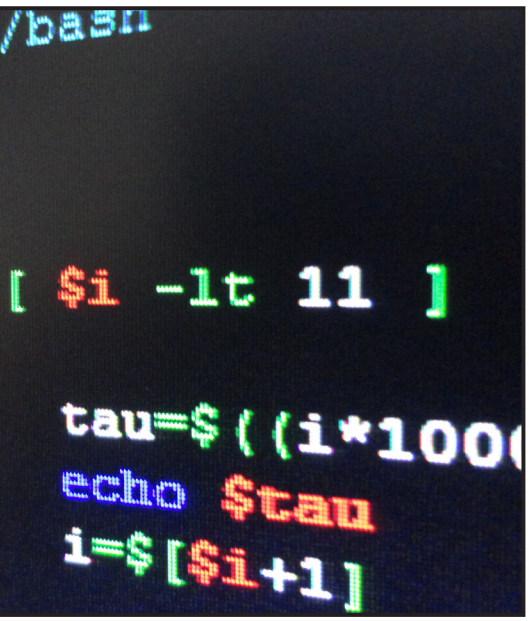
```
$ sudo chmod +x meow.sh
```

**3. Ahora ejectua el script escribiendo:**

```
$ ./meow.sh
```

Debe aparecer "miau". ¡Bash es ahora un gato robot! Ahora, vamos a modificar nuestro script para cambiar lo que puede decir nuestro gato. Quizás diga algo positivo o algo ofensivo, la cuestión es que puedes modificarlo. Modifica meow.sh con:

```
#!/bin/bash
uno=${1:-"me"} #use first param-
```




```
eter supplied, else use default
provided
dos=${2:-"ow"}  #use second param
supplied, else use default pro-
vided
echo "${uno}${dos}" #nothing new
here…
```

Ahora puedes escribir `./meow.sh` y ver el comportamiento del gato del ejemplo anterior, o bien puede escribir `./meow.sh <cualquier cosas>` y el gato robot lo repetirá. Por ejemplo, supón que tu gato ve a otro atractivo gato robot y quiere llamar su atención. Escribe `./meow.sh meeeeeee owww` e inicia una conversación.

## Automatización

Supongamos que realmente quiere hacer algo útil con bash, como escribir un script para ejecutar unos cuantos comandos cada vez que arranque tu ODROID. Yo uso el siguiente script, por ejemplo, para iniciar un servidor TCP en cada U3+ de mi clúster para responder a las solicitudes de trabajo de los clientes:

**1. Crea un archivo llamado on_boot.sh:**

```
#!/bin/bash
sleep 15   # pause execution for
15 seconds
cd /home/of/your/file    # change
places
$ twistd -y WorkNode.tac -l logs/
node.log   # put your command here
```

**2. Ahora edita `/etc/rc.local` y añade las siguientes líneas antes del 0:**

```
cd /the/folder/where/on_boot/
lives/
./on_boot.sh
```

Además, no olvides hacer tu script ejecutable con `sudo chmod +x` on_boot.sh.

## Hacer algún cálculo

Bash también puede hacer cosas útiles como ejecutar bucles y hacer cálculos. Crea un archivo llamado math_loop.sh:

```
#!/bin/bash
for i in 1 2 3 4 5 6 7 8 9 10
do
      tau=$((i*1000))   # aka tau
is i * 1000
      echo tau
done
```

Tras darle los mismos permisos de archivo que el anterior, el script monstrará 1000, 2000, ..., 10000 en tu consola. Otra posibilidad es ejecutar el script usando un bucle. Como es de esperar, los operadores de comparación son algo diferentes en bash:

```
#!/bin/bash
i=1
while [ $i -lt 11 ]   # -lt means
less than
do
        tau=$((i*1000))
        echo $tau
        i=$[$i+1]   # increments
i by 1
done
```

Ambos programas tienen el mismo comportamiento, pero trabajan de un modo distinto. Ahora vamos a ponerlo todo junto y crear un script para ejecutar múltiples script en un bucle. Utilizo este script siempre que necesito enviar manualmente varias solicitudes de trabajo a mi clúter:

```
#!/bin/bash
symbol=$1
date=$2
for i in 1 2 3 4 5 6 7 8 9 10
do
        ip=$((i-1))
        tau=$((i*1000))
        path="${symbol}_${date}"
        python WorkClient.py -t
$tau -f $path -i 192.1.1.$ip
done
```

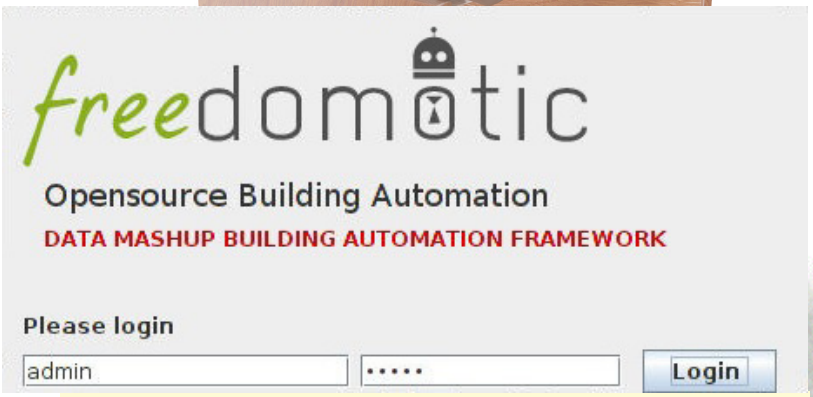
Este script coge dos argumentos, el nombre del símbolo y la fecha en formato AAAAMMDD, y los utiliza para enviar trabajos utilizando WorkClient.py (un cliente TCP). Observa cómo el script bash crea los parámetros ip, path y tau que se pasan a WorkClient.py. Bash es una poderosa herramienta que todo administrador ODROID debe tener cerca. En este artículo he presentado algunas formas muy básicas de cómo podemos utilizar bash, incluidas operaciones simples de cálculo, la combinación de cadenas y bucles. Añadir argumentos a tus scripts expande efectivamente el número de funciones que puede realizar. Si eres como yo y quieres automatizar tanto flujo de trabajo como te sea posible, bash puede ayudarle a liberarte de muchas tareas tediosas que pueden automatizarse, así como proporcionarte una herramienta rápida y potente para interactuar con tus sistemas.

Para descargar el código de ejemplo utilizado en este artículo visita www.odroidcluster.com y puedes enviarnos cualquier pregunta, error o sugerencia a odroidcluster (at) gmail.com.

# INSTALACION DE FREEDOMOTIC

## UN ENTORNO DE TRABAJO PARA LA AUTOMATIZACION DE EDIFICIOS

**Por Venkat Bommakanti**




**Pantalla de inicio del software de código abierto Freedomotic**

Te interesa la Domótica. Los sistemas ODROID son grandes controladores y gestores de automatización! Pueden ser utilizados para controlar tu casa a distancia, o establecer tareas que se realizan de forma automática cada día. En este artículo se describe cómo usar del U3 para montar un sistema automatizado en una oficina, casa o edificio usando el software Freedomotic.

## Requisitos:

**-Una placa ODROID U3 con el adaptador de alimentación adecuado .**
**-Una tarjeta MicroSD de 8+ GB o módulo eMMC que contenga la imagen de Lubuntu para U3 más reciente disponibles en la web de Hardkernel.**
**-Una red en la que el dispositivo tenga acceso a internet y a los foros ODROID.**
**-Acceso SSH opcional al U3 con utilidades como PuTTY (MS Windows 7 +) o Terminal (Mac, Linux) para realizar los pasos desde un ordenador remoto.**

## Instalar Apache maven

Apache Maven es una herramienta de comprensión y gestión de proyectos de software. El software freedomotic utiliza esta infraestructura. Se puede instalar con el comando:

```
$ sudo apt-get install maven
```

## Código fuente de freedomotic

Puesto que no existen paquetes de ubuntu basados en ARM pre-compilados, tendrás que crear el entorno de trabajo desde su código fuente en el U3. Este artículo no aborda la compilación cruzada.

Crea un subdirectorio para recibir las fuentes y cámbiate a éste, con los comandos:

```
$ mkdir freedomotic-src
$ cd freedomotic-src/
```

Traslada el código fuente desde el repositorio git pertinente a este ubicación, con el comando:

```
$ git clone https://github.com/
freedomotic/freedomotic.git
```

## Compilar freedomotic usando Maven

Un nuevo subdirectorio con el árbol completo de fuentes se crea automáticamente. Navega hasta éste y lanza el proceso de compilación con los comandos:

```
$ cd freedomotic/
$ mvn clean install
```

Una vez finalizada la compilación, Maven puede iniciarse usando el comando mvn.

## Instalar los datos de ejemplo

Crea una copia de los datos de ejemplo utilizando el comando:

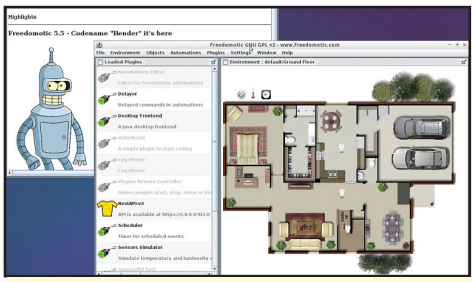
```
$ cp -r data-example/ \
framework/freedomotic-core/data
```

## Ejecutar freedomotic

Lanza la plataforma utilizando el comando:

```
$ java -jar framework/\
freedomotic-core /target/\
freedomotic-core/freedomotic.jar
```

A continuación, aparecerá un cuadro de diálogo de inicio de sesión.



Pantalla de bienvenida de Freedomotic

Utiliza el usuario y contraseña admin/admin para iniciar sesión. ¡Ahora el mundo de la automatización esta en tus manos! Asegúrese de investigar bien el tema de la automatización y tomar todas las precauciones a la hora de automatizar y hacer público cualquier aspecto de tu vida, incluyendo la domótica y el acceso vía web. La información que muestra este articulo sólo tiene fines educativos y de entretenimiento.

## Plugins

Freedomotic funciona con plugins. Puede usar plugins gratuitos de código abierto o desarrollar los tuyos propios. El repositorio GIT contiene el SDK completo que tiene todo el código que necesitas para desarrollar y probar tus propios plugins. Después de compilarlo por primera vez, abre el proyecto freedomotic con tu IDE favorito.

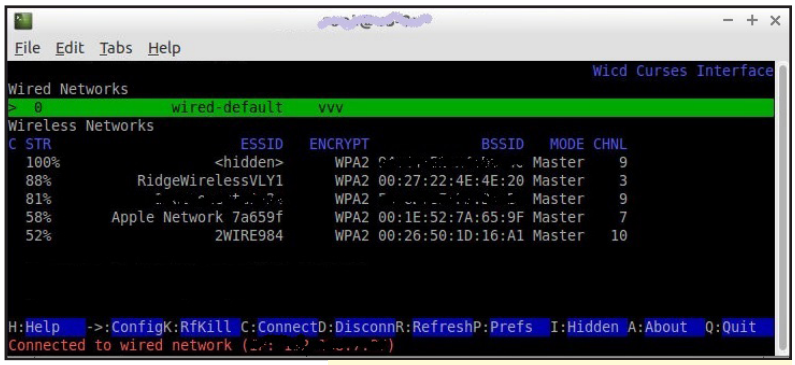
Para desarrollar su propio plugin puedes empezar con el proyecto de ejemplo hello-world incluido en el directorio plugins/devices/hello-world. Abrelo en tu IDE, haz algunos cambios y compilalo. Se instalará automáticamente en el proyecto freedomotic. Simplemente reinicie freedomotic para probar tus últimos cambios.

Para obtener información adicional o realizar cualquier pregunta, visita las fuentes de información originales en http://bit.ly/1qqjyun, http://bit.ly/1nL16ZI y http://bit.ly/1Cdwdai.

# INSTALACION DE WICD

## UN GESTOR DE CONEXIONES DE RED

**Por Venkat Bommakanti**




WICD permite al ODROID conectarse fácilmente a cualquier red por cable o inalámbrica

Si deseas utilizar un gestor de conexiones para gestionar interfaces de red por cable y WiFi, la utilidad liviana wicd es una de las posibles opciones. Es una alternativa a la herramienta Network Manager basada en gnome. En este artículo se describe cómo instalar wicd en ODROID U3.

## Requisitos

**1. Una placa ODROID U3 con el adaptador de alimentación adecuado.**
**2. Una tarjeta MicroSD (con un lector de tarjeta SD) que contenga la última imagen del escritorio de Lubuntu específica para U3, o un módulo de eMMC 8+ GB.**
**3. Una red en la que el dispositivo tenga acceso a internet y a los foros ODROID.**
**4. acceso SSH al U3 a través de utilidades SSH como PuTTY (MS Windows 7 +) o Terminal (Mac, Linux) desde el escritorio remoto.**

## Instalar wicd y la infraestructura necesaria

Ejecuta el siguiente comando para instalar los componentes necesarios:

```
$ sudo  apt-get install \
wicd-curses wicd
```

## Iniciar la aplicación y el servicio wicd

Inicia la aplicación y el servicio requerido, con el comando:

```
$ sudo service wicd
start
 * Starting Network connection
manager wicd
 [ OK ]
$ sudo wicd-curses
```

## Verificar la instalación

Ha de aparecer una interfaz de usuario con una lista de redes, como la que aparece en la imagen de arriba.

Consulte las páginas guía o la ayuda en línea para obtener detalles de su uso. Para información adicional o preguntas, visite las fuentes de información originales en http://bit.ly/1powWRH y http://bit.ly/1vTU7Df.

# 3DPONICS

## UN SISTEMA DE JARDINERIA DE CODIGO ABIERTO BASADO EN ODROID

*Por Lucy Morrissey*

Eres una persona de ciudad que carece de espacio para un jardín, un amante de los alimentos frescos cansado de pagar una fortuna por la verdura en las tiendas, o simplemente estas demasiado ocupado para cultivar tus verduras. Ahora, puedes cultivar tu propia comida en casa usando un sistema de jardinería de última generación llamado 3Dponics, disponible gratuitamente en http://www.3Dponics.com. ¡Lo mejor de todo es que puedes fabricar (o mejor dicho imprimir en 3D) la mayoría de los componentes necesarios desde casa!

Durante los últimos dos años, 3Dprintler, un laboratorio de tecnología con sede en Ottawa, ha desarrollado un sistema de cultivo hidropónico en 3D y se está ofreciendo al público como un proyecto completamente de código abierto. Debido a su bajo precio y potente procesador, 3Dponics eligió recientemente la ODROID-U3 como hardware para la conexión de los sensores de jardín vía Internet, pudiendo gestionar y controlar el jardín de forma remota.

**Con 3Dponics, las verduras se pueden cultivar en casi cualquier espacio de reducidas dimensiones**

## Comencemos

**1. Descarga los archivos desde la web de 3Dponics o desde la cuenta 3Dprintler en http://www.thingiverse.com.**
**2. Imprime en 3D los archivos, o usa un servicio de impresión 3D para crear los componentes del sistema.**
**3. Reúne las piezas que no se pueden imprimir en 3D de tu hogar o acude a una ferretería.**
**4. Configura el sistema, siguiendo las instrucciones y los sencillos video tutoriales disponibles en el sitio web 3Dponics..**

## Software y hardware

Se usaron varios tipos de software, SolidWorks, AutoCAD, SketchUp para diseñar los componentes 3Dponics y prepararlos para su impresión. Aunque 3Dponics necesita el software para crear los archivos, tú no lo necesitas, ya que puede acceder a los archivos ya creados vía on-line. No obstante puedes modificarlos y compartir tus cambios con otros usuarios.

Cuando los archivos originales fueron diseñados y preparados, las partes se imprimieron en 3D utilizando un Makerbot Fifth Generation Replicator y un Formlabs Form 1+. Todo el sistema podía imprimirse en tan sólo cinco horas a una baja resolución.

Si no dispones de a una impresora 3D, todavía puedes montar el sistema 3Dponics contactando con el servicio de localización de impresoras 3D en http://www.3dhubs.com. Ellos te conectaran con alguien próximo que tenga una impresora 3D y podrás imprimir los archivos. Con este servicio, no es necesario pagar los gastos de envío y podras tener listos los componentes en poco tiempo. Las partes más importantes son la boquilla de goteo, tuberías y silenciador.

## Componentes a imprimir

**Las boquillas de goteo para las botellas de plástico**
**Tubería con un agujero para el acuario**
**Sifón para el agua**
**Tubo para la bomba de aire del acuario**
**Silenciador para reducir los niveles de ruido**
**Tapón exterior para botella**
**Tapón interior para botella**

El jardín 3Dponics puede utilizar cosas baratas del hogar

Regulador superior de la botella
Conector de la bomba para múltiples sistemas
Barra de soporte modular para tuberías
Funda protectora para botellas
Aspersor para las plantas
Aspersor para las raíces

## Componentes no imprimibles

3-4 botellas de plástico vacías (recomendado de IL o 2L)
Bomba de aire silenciosa Hagen Marina 200 (o equivalente)
3,5 metros de tubería para la fuente de aire del acuario
20 bandas de sujeción

## Sistema autónomo

Gracias a la ODROID-U3, el sistema 3Dponics se puede activar por Internet. Después de experimentar con diferentes unidades, se descubrió que el ODROID-U3 era el que mejor trabajaba con el sistema de 3Dponics porque:

1.El cote del ODROID U3 (65$) es menos que el de otras placas como Intel Nuc i3 ($ 300), y las tarjetas microSD, cámaras HD USB y sensores de datos son muy económicos.
2. El sistema operativo (SO) es gratuito (Android o Linux),
3. Puedes imprimir en 3D tu propia caja a medida para ODROID-U3 en http://bit.ly/1qmCfAv (gracias al usuario de Thingiverse, miguif).

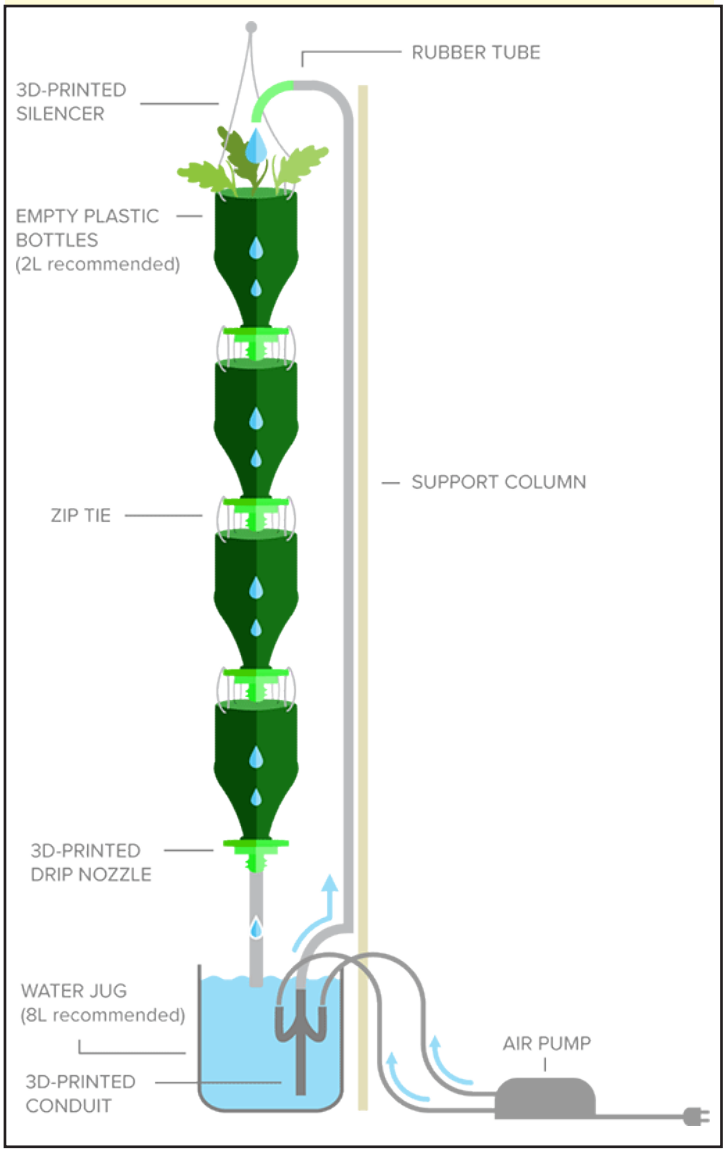
## Aplicación ODROID-U3

Hay una app llamada 3Dponics Farm App en continuo desarrollo y que los usuarios pueden instalar en su propio dispositivo ODROID U3. La app coge los datos de los sensores del sistema 3Dponics y se comunica con los servidores 3Dponics, que envían los datos a un teléfono inteligente.

Los usuarios simplemente abren la app en su teléfono (Android o iOS) y se conectan con el servidor ODROID-U3 para monitorear y controlar su propio sistema 3Dponics. Pueden comprobar la temperatura y humedad, encender y apagar el sistema, configurar un temporizador (por ejemplo, para programar una acción cuando los índices sean muy bajos), ver la señal de video en vivo, sincronizar el sistema con el amanecer y el atardecer, y conectarlo a paneles solares como una fuente de energía renovable.

Puesto que el espacio usado por el sistema es pequeño, sólo requiere 4,5 vatios de electricidad para hacerlo funcionar. Añadiendo células solares y una batería, la bomba de aire y el ODROID-U3 pueden ser alimentados totalmente con luz solar.

Un típico diseño de un sistema de jardinería exterior 3Dponics



# WALL-E

## CONSTRUYE TU PROPIO ROBOT EN CASA PARTE I

*Por Vincenzo Siriaani*


El modelo básico de control remoto Wall-E de Pixar está disponible para su compra en Amazon, listo para usar con un ODROID U3 como cerebro!

Siendo un niño tuve un sueño, deseaba tener un robot en casa. Recientemente y puesto que la robótica domestica se ha vuelto más asequible, comencé a programar robots con Arduino creando algunos para caminar y evitar obstáculos, pero eran robots básicos sin ninguna personalidad. Hace unos 6 meses, empecé a estudiar Python, puesto que Java y C eran demasiado difíciles para mí.

Compré un ODROID U3 con un módulo eMMC con Linux, a pesar de que nunca antes había usado Linux. De hecho, ni siquiera pude instalar la librería básica OpenCV. Con el tiempo descargue una imagen SO de los foros ODROID de Robótica llamada Linaro 12.11 Robotics Edition (ROS), y usé código Python para enseñar a mi robot a detectar caras. Después llegué a trabajar con OpenCV, conecté un Arduino al U3 usando un cable USB y envié algunas secuencias de código.

La inspiración para mi robot Wall-E empezó cuando vi la película "Wall-E", entonces me compre un Wall-E U-Command usado, fabricado por Pixar (http://amzn.to/1lBYyC2). En mi búsqueda, encontré el sitio de DJ Sures (http://bit.ly/1pfKxEQ) y me di cuenta que era posible hacer el mismo robot sin un PC con Windows, utilizando para ello un ODROID-U3

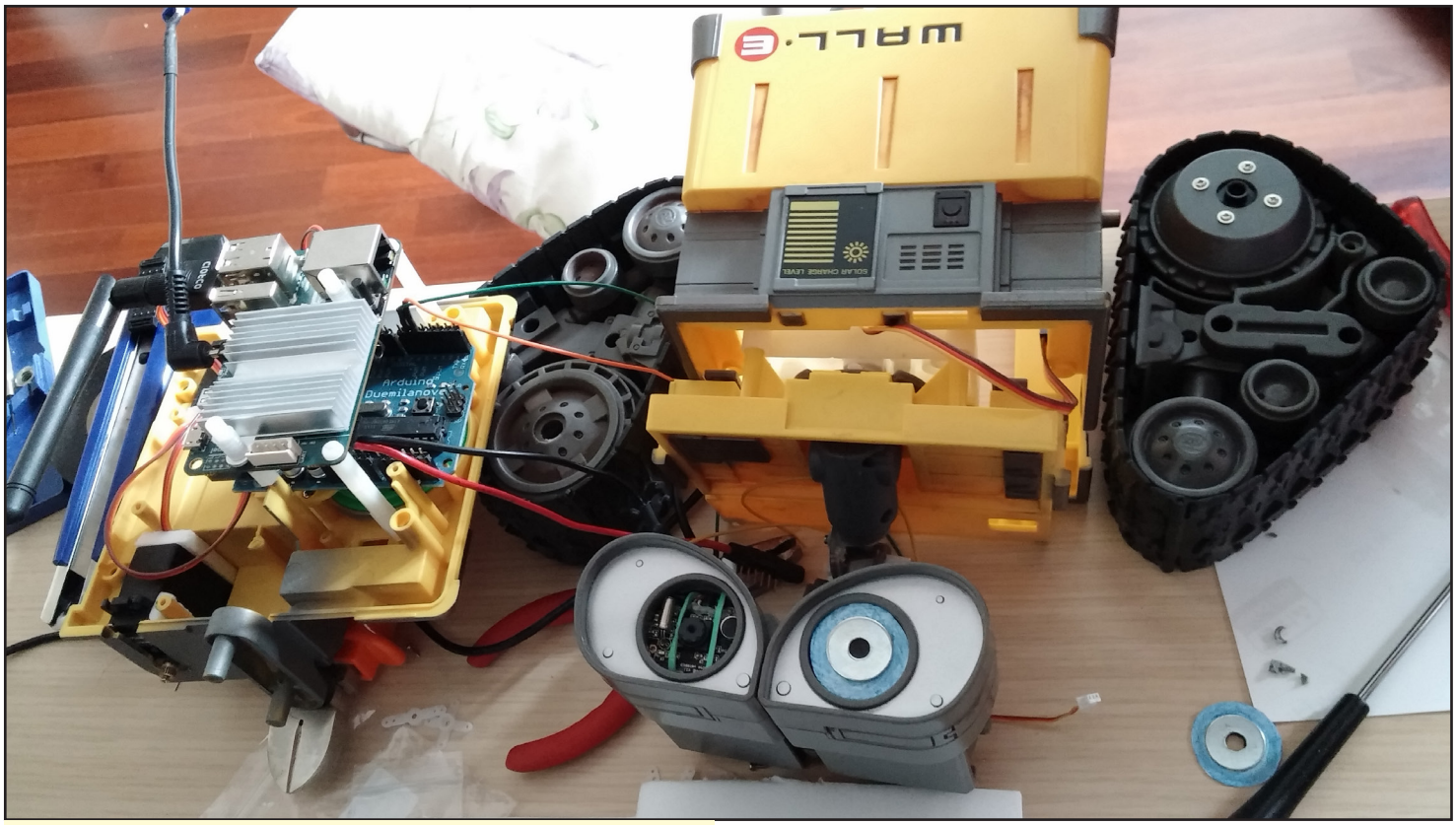
## Materials

1. Arduino 2009
2. ODROID U3
3. Webcam de Hardkernel
4. eMMC con Ubuntu preinstalado
5. Tarjeta MicroSD clase 10 (para Linaro)
6. 2 servomotores con engranajes de metal para el "gusano" - Servo rodamientos Turnigy Digital de gran potencia 26,0g/3,5 kg/0.12sec
7. 2 servomotores para la vista panorámica y la inclinación – Micro servomotores analógicos HK15168 8g/1.2kg/ 0.12s Micro Servo 8g / 1.2kg / 0.12s
8. 2 Ultramicro servomotores para los ojos - HK-282A de un único tornillo, Ultra-Micro Servomotores 2g/0,2 kg/0.08sec
9. 2 servomotores para los brazos - Micro servomotores analógicos HK15168 8g/1.2kg/0.12s
10. Un regulador de tensión de 5 voltios (máximo 5 Ah) para alimentar los circuitos - TURNIGY 3A UBEC w/Reducción del ruido
11. Batería Lipo - Turnigy nano-tech A-SPEC 2200mAh 3S 65 ~ 130C Lipo Pack

Desmonte el robot Wall-E que compré e inserté los servomotores principales y webcam, como se muestra en las fotos de este. Luego, empecé a desarrollar los controladores de software. Por ahora, he escrito el código en Python 2.7, sin ningún GUI. Parte del código es para la detección de rostros y el seguimiento, otra parte es para enviar el vídeo a otro equipo y otra para recibir los comandos.

**¡Wall-E rescató a un amigo y decidió resucitarlo como un ODROID!**

```
# Wall-E Main Controller Python Script

import threading
import time
import cv2
import timeit
import socket
import serial
import numpy
# these are all the libraries
global command
global face_positionx
global face_positiony
#queue
face_positionx = 0
face_positiony = 0
command = “”

class tasks(threading.Thread):
    def __init__(self, threadID, name, counter, func-
tions):
        threading.Thread.__init__(self)
        self.threadID = threadID
        self.name = name
        self.counter = counter
        self.functions = functions
    def run(self):
        print “Starting “ + self.name
        self.functions()
        print “Exiting “ + self.name
# this is a class that performs the server that re-
ceives the command, i don’t know why, but
# without this class the thread doesn’t works

def server():
    global command
    server_socket = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
    server_socket.setsockopt(socket.SOL_SOCKET, sock-
et.SO_REUSEADDR, 1)
    server_socket.bind((“localhost”,5001))
    server_socket.listen(5)
    client_socket, address = server_socket.accept()
    while 1:
        client_command = client_socket.recv(1024)
        client_socket.send(“From “ + repr(address) +
“ Recived “ + repr(client_command))
                command = client_command
        if client_command == “q”:
            time.sleep(5)
            client_socket.close()
            server_socket.close()
            break
    print “Uscito server”

# this is a server that receives the command for the
Wall-E, you can change the “localhost”
# with a ip number and it works out of the same ma-
chine.  to send the command
# out of the home network you must to redirect the
port in the settings of your router

def cattura_immagine():
    time.sleep(1)
    global command
    global stringData
```

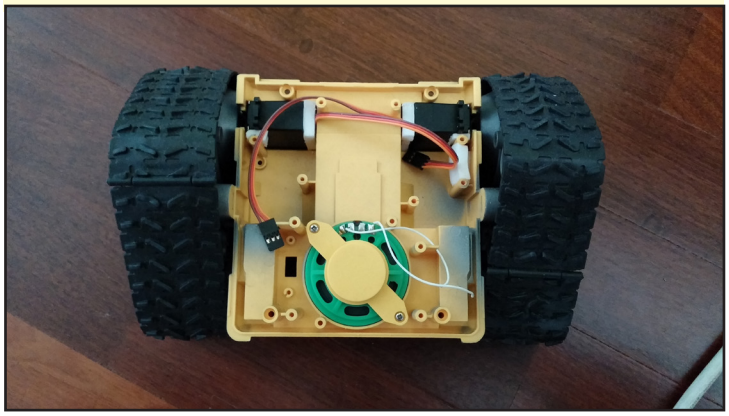


**Partes de Wall-E, desmontadas para probar los motores del ojo**

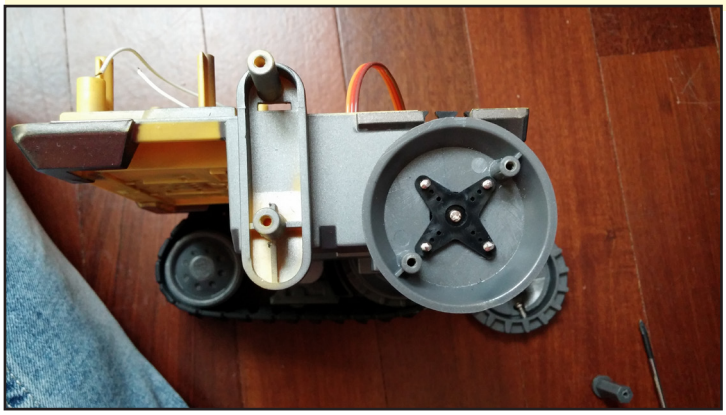
```
    face_positionx = 0
    face_positiony = 0
    command_arduino = 0
    TCP_IP = "192.168.1.107"
    TCP_PORT = 5002
    print "before socket.socket"
    sock = socket.socket(socket.AF_INET, socket.SOCK_
STREAM)
    sock.setsockopt(socket.SOL_SOCKET, socket.SO_RE-
USEADDR, 1)
    print "after socket.socket"
    sock.bind((TCP_IP, TCP_PORT))
    sock.listen(5)
    client_socket_video, address = sock.accept()
    # this part of code start the variables and video
serve, you can change the IP
    # number as the server for commands

    print "ARDU"
    try:
        arduino = serial.Serial('/dev/tty-
USB0',115200)
        arduinoconnesso = 1
        print "ARDUINO CONNESSO"

    except:
        arduinoconnesso = 0
        print "ARDUINO NON CONNESSO"
    # this piece try to connect the ODROID U3 to the
arduino, you can change the port name and
    # the speed

    SCALA = 2
    TRAINSET = "/home/linaro/opencv-2.4.6.1/data/lbp-
cascades/lbpcascade_frontalface.xml"
    classifier = cv2.CascadeClassifier(TRAINSET)
    # this set the cascade for the search of faces,
lbp is faster, but you can change

    webcam = cv2.VideoCapture(0)
    time.sleep(3)
    if webcam.isOpened():
        print "Video aperto"
        ret, frame = webcam.read()
        if ret:
            print "ret True"
            contatore = 0
            # this start the webcam capture, you can
change the number of the webcam port

            while(1):
                contatore = contatore + 1
                ret, frame = webcam.read()
                t = timeit.default_timer()
                height = frame.shape[0]
                width = frame.shape[1]
```

```
                cv2.putText(frame, "Larghezza " +
repr(width) +
                            "Altezza " + repr(height),
(50,10),
                             cv2.FONT_HERSHEY_SIMPLEX,
0.6, (255,255,255))
                minisize = (frame.shape[1]/
SCALA,frame.shape[0]/SCALA)
                miniframe = cv2.resize(frame, mini-
size)
                gray = cv2.cvtColor(miniframe, cv2.
COLOR_BGR2GRAY)
                gray = cv2.equalizeHist(gray)

                # all this part of code starts to
read frames for ever, gets the time for FPS,
                # shrinks the frame, turns the frame
first to grey scala, then makes Histogram
                # Equalization, to improves the con-
trast in the image, to speed up the process
                # of face detection

                if command == "4":
                    command_arduino = "4"

                if command == "5":
                    command_arduino = "3"

                if command == "r":
                    command_arduino = "2"

                if command == "t":
                    command_arduino = "1"
                if command_arduino != 0:
                    if arduinoconnesso == 1:
                        arduino.write (repr(command_
arduino))
                    command_arduino = 0
                    command = ""

            # this is for the manual command of pan
and tilt of Wall-E head

                if command == "f":
                    faces = classifier.
detectMultiScale(gray)
                    for f in faces:
                        x, y, w, h = [ v*SCALA for v
in f ]
                        cv2.rectangle(frame, (x,y),
(x+w,y+h), (0,0,255))
                        cv2.rectangle(frame, (x+w/2-
1,y+h/2-1), (x+w/2+1,y+h/2+1), (0,0,255))
                        cv2.putText(frame, "X =
"+repr(x+w/2)+" Y = " + repr(y+h/2), (5, 25),
                            cv2.FONT_HERSHEY_SIMPLEX,
0.6, (255,255,255))
                        face_positionx = repr(x+w/2)
                        face_positiony = repr(y+h/2)
                        cv2.putText(frame, "Volti n.
" + repr(len(faces)), (x-50,y-10),
                            cv2.FONT_HERSHEY_SIMPLEX,
0.6, (255,255,255))

# this performs the real face detection,
                    # (http://bytefish.de/blog/
opencv/object_detection/)

                    if face_positionx != 0 or face_
positiony != 0:
                        if int(face_positionx) < 220:
                            command_arduino = "4"
                        if int(face_positionx) > 390:
                            command_arduino = "3"
                        if command_arduino != 0:
                            if arduinoconnesso == 1:
```

**Una vista frontal de las ruedas tractor de Wall-E y el motor**



**Una vista lateral de las ruedas tractor de Wall-E sin las bandas**



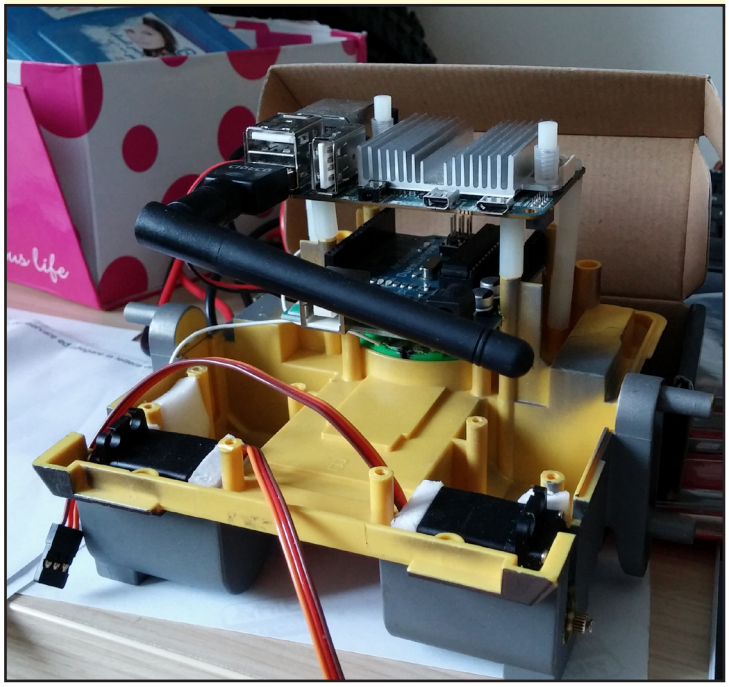
**Podemos contemplar una caja impermeable para el U3, ya que a Wall-E le encanta nadar!!!**

```
                                arduino.write
(repr(command_arduino))
                            command_arduino = 0

                        if int(face_positiony) < 160:
                            command_arduino = "1"
                        if int(face_positiony) > 320:
                            command_arduino = "2"
                        if command_arduino != 0:
                            if arduinoconnesso == 1:
                                arduino.write
(repr(command_arduino))
                            command_arduino = 0
                    face_positionx = 0
                    face_positiony = 0
                dt = timeit.default_timer() - t
                cv2.putText(frame, "FPS =  " +
repr(1/dt), (5, 50),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.6,
(255,255,255))

          # this part sends the command to Wall-E
head to track the faces

                encode_param = [int(cv2.IMWRITE_JPEG_
QUALITY),90]
                result, imgencode = cv2.imencode('.
jpg', gray, encode_param)
                data = numpy.array(imgencode)
                stringData = data.tostring()
                if contatore == 3:
                    client_socket_video.
send(str(len(stringData)).ljust(16));
                    client_socket_video.
send(stringData);
                    contatore = 0
          # this send the stream video to the client
video, it send strings of text!!
                cv2.imshow('frame',frame)
                if command == "q":
                    webcam.release()
                    cv2.destroyAllWindows()
                    break
                if cv2.waitKey(1) & 0xFF == ord('q'):
                    command = "q"
                    break

            sock.close()
            time.sleep(2)
            if arduinoconnesso == 1:
                arduino.close()
            webcam.release()
            time.sleep(2)
            cv2.destroyAllWindows()
            print "Uscito Opencv"

            time.sleep(2)
               # this closes all when you push "q"

# this starts everything
print "Comincio"
thread2 = tasks(2, "server", 2, server)
thread2.start()
cattura_immagine()
```

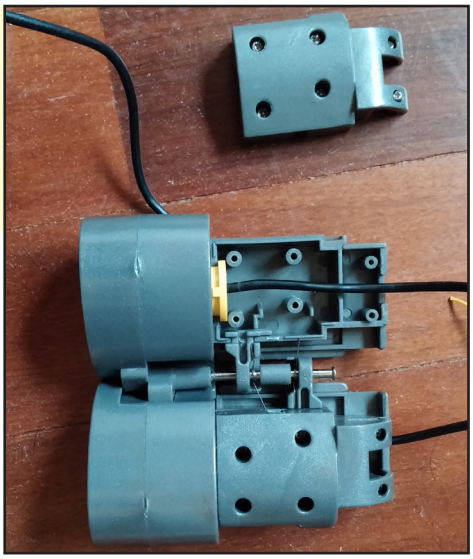
El script anterior es el software principal, pero hay otros 2 necesarios. Tras iniciar el  módulo principal, el servicio de comandos central y el servicio de video se ponen en marcha:

```
import socket                    # Import socket module

s = socket.socket()              # Create a socket object
host = "localhost"               # Get local machine name
port = 5001                      # Reserve a port for
your command service

s.connect((host, port))
print "Connect to " + host
print "Use f for detection, 4, 5, r, t for manual
command of the head, q to quit "
while (True):
    command = raw_input ("Command? ")
    s.send(command)
    print s.recv(1024)
    if command == "q":
        break
s.close                          # Close the socket when
"q"

video client:

import socket
import cv2
import numpy

def recvall(sock, count):
    buf = b''
    while count:
        newbuf = sock.recv(count)
        if not newbuf: return None
        buf += newbuf
        count -= len(newbuf)
    return buf
# http://stupidpythonideas.blogspot.it/2013/05/sock-
ets-are-byte-streams-not-message.html

TCP_IP = "192.168.1.107"
TCP_PORT = 5002

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((TCP_IP, TCP_PORT))
while (True):

    length = recvall(s,16)
    stringData = recvall(s, int(length))
    data = numpy.fromstring(stringData,
dtype='uint8')

    decimg=cv2.imdecode(data,1)
    cv2.imshow('SERVER',decimg)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
s.close()
```

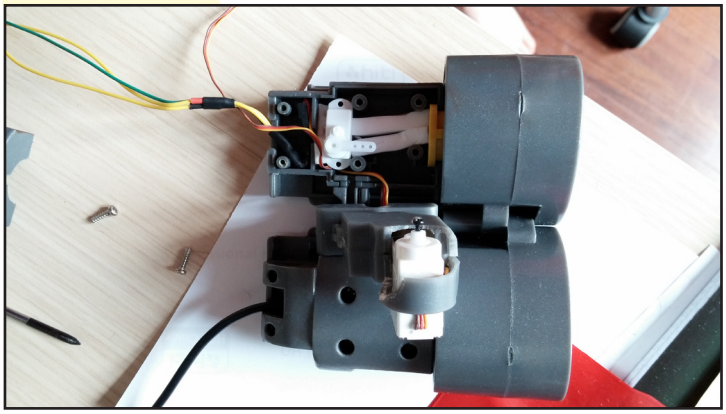
**ODROID-U3 montado en la base Wall-E, con el dongle inalámbrico**

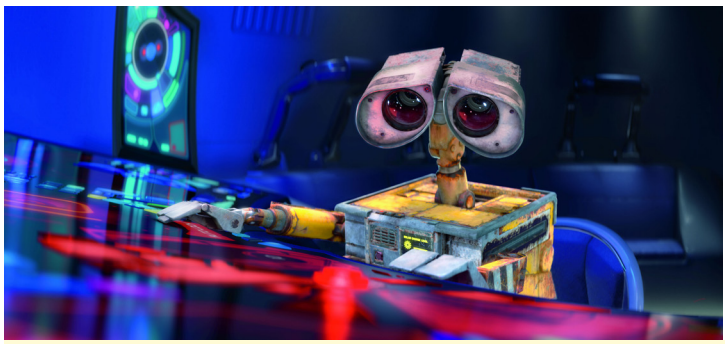


**Primer plano de los servomotores utilizados para controlar los movimientos del ojo de Wall-E**



**Los ojos de Wall-E son su carta de presentación y tienen que ser muy expresivos**





**Wall-E utiliza una pantalla táctil ODROID-VU para programar a su nuevo colega**

```
cv2.destroyAllWindows()
# http://stackoverflow.com/questions/20820602/image-
send-via-tcp

The Arduino code is very simple, and does noth-
ing more than receiving the messages from ODROID and
sending them to the servos.  Note that the servos of
the head are connected to pin 5 and 6.

#include <Servo.h>

Servo myservo1;
Servo myservo2;

byte rx = 0; // variabile per contenere il carattere
ricevuto.
int pan = 90;
int tilt = 77;

void setup()
  {
Serial.begin(115200); // imposto la seriale per
lavorare a 115200 baud
myservo1.attach(6);
myservo1.write(pan);
myservo2.attach(5);
myservo2.write(tilt);
Serial.flush(); // svuoto il buffer di ricezione se-
riale.
delay(100);
  }

void loop()
  {

  if (Serial.available() >0) // Controllo se il buf-
fer di ricezione contiene qualcosa
    {
      rx = Serial.read(); // leggo il carattere
ricevuto e lo memorizzo in rx
      Serial.flush(); // svuoto il buffer di ricezione
seriale
      if (rx != '0')
       {
        if (rx=='1')
          {
            if (pan >= 35)
            {
              pan = pan - 2;
            }
          }
        if (rx=='2')
          {
            if (pan <= 135)
            {
              pan = pan + 2;
            }
          }
        if (rx=='3')
          {
            if (tilt >= 35)
            {
              tilt = tilt - 2;
            }
          }
        if (rx=='4')
          {
            if (tilt <= 125)
            {
              tilt = tilt + 2;
            }
          }
       }
        myservo1.write(pan);
        myservo2.write(tilt);
  }  }
```

Mi siguiente paso, añadir el habla y el reconocimiento de voz a Wall-E, y luego conectar los servomotores a Arduino para que mi robot tenga móvilidad.

**Wall-E vio el artículo 3Dponics y puso en marcha su propio jardín!**

# EL PARTE METEOROLOGICO EN EL ESCRITORIO

## ¿HAY POSIBILIDAD DE PESCAR EL PROXIMO FIN DE SEMANA?

**Por Jussi Opas**

Hay algún plan para ir a pescar el próximo fin de semana, jugar al golf, o simplemente quedarnos en casa y usar el ordenador? Esta es una pregunta que los aficionados al aire libre están siempre repitiendo durante la semana. Para ayudar a responder esta cuestión, los usuarios de Linux pueden instalar una previsión del tiempo directamente en el escritorio. En este artículo, te presento un par de aplicaciones que muestran el tiempo actual y el pronóstico para ayudarte a planificar futuras actividades al aire libre.

### XFCE

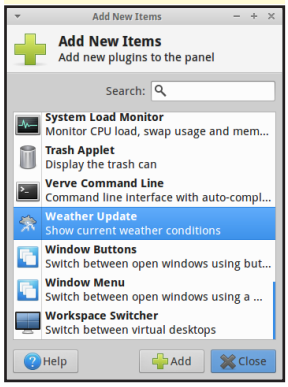
El escritorio XFCE tiene un plugin del tiempo y se pue-de añadir al panel inferior. Se puede agregar un widget de Weather Update, como se muestra en la cimagen de la derecha. Para instalar el plugin del tiempo escribe:

```
$ sudo apt-get install xfce4-weather-plugin
```

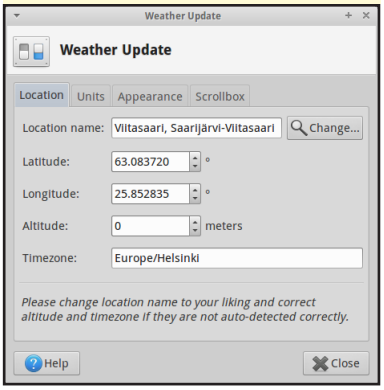
Desde el panel, se puede abrir las propiedades haciendo clic derecho para acceder a las opciones de configuración..

El botón Change abre un segundo cuadro de diálogo para seleccionar una ubicación. La ciudad más cercana a nuestro río de pesca es “Viitasaari”, que puede ser localizado introduciéndolo en el cuadro de texto y pulsando el botón Search. De la lista de resultados, se puede seleccionar la localización deseada y obtener su pronóstico.

**Añadiendo Weather Update**




**Diálogo de configuración de la aplicación Weather Update**




Al pasar por encima del widget del tiempo con el ratón se muestra un texto con información de la temperatura, el viento, la humedad y la nubosidad. Al hacer clic en el widget, se puede obtener un pronóstico para los próximos días.

Para atrapar truchas, el tiempo debe estar nublado o algo lluvioso, más que soleado. Las truchas aparecen a última hora de la tarde o en las primeras horas de la noche. Por otro lado, el viento no es bueno para la pesca con mosca, ya que suele afectar a la capacidad de lanzar el anzuelo. Además de que no se puede ver la actividad de los peces en el agua cuando hace demasiado viento.

**En este fondo de escritorio de http://bit.ly/IIGnGrj, se ve un martín pescador como fondo de pantalla. ¿Será la predicción de nuestra oportunidad para pescar?**

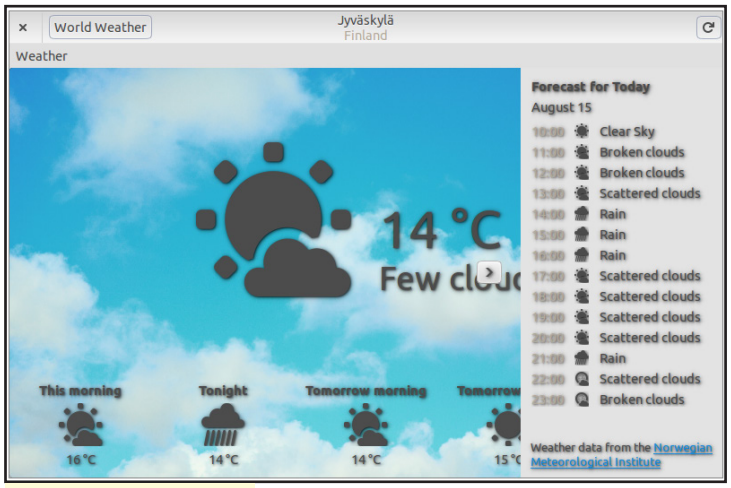
Varios widgets como el viento y la temperatura se pueden añadir para ayudarnos a decidir cuándo ir a pescar. En el ejemplo que se muestra en la captura de pantalla, los días de pesca seleccionados son el sábado y el domingo, y parece que el mejor tiempo para pescar tendrá lugar el sábado al anocher, ya que no hay prácticamente nada de viento, estará algo nublado y oscuro. El domingo por la mañana también nos puede interesar, ya que parece que estará nublado.

### Gnome y Unity

Los usuarios de casi todos los escritorios, incluyendo Gnome y Unity, también tienen aplicaciones meteorológicas disponibles en diversas fuentes de software. Busca por “weath-

er" en el Centro de Software de Lubuntu o Ubuntu. Por ejemplo, gnome-weather está disponible en el Centro de Software de Lubuntu, como se muestra en la siguiente captura de pantalla y puede ser instalado escribiendo:

```
$ sudo apt-get install gnome-weather
```




**Gnome Weather**

Usando gnome-weather, el usuario puede elegir un determinado lugar y ver el tiempo para esa zona. Se puede ver el pronóstico del tiempo para el día en curso y también una predicción bastante precisa para la siguiente semana.

## Salir de Pesca

Continuando con nuestra historia, llegó el fin de semana y fuimos al río, basándonos en la información que teníamos de nuestro widget del tiempo. El cielo estaba nublado el sábado por la tarde. El domingo por la mañana en lugar de estar nublado, estaba soleado y sólo aparecieron unas cuantas nubes. El tiempo previsto tuvo lugar pero sucedió antes de lo esperado. Es posible que que las masas de aire se movieran más rápido de lo esperado.

El sábado al anochecer estaba nublado y justo antes de una li-gera lluvia, los peces parecian estar activos y respondieron a la serpentina oropel. Por desgracia, el pez cazado se escapó cuando se rompió el cable.

Pronto oscureció, así que tuvimos que dejar de pescar. Era mejor irse a dormir y pescar de nuevo el domingo por la mañana. Usamos el oropel porque tuvimos algo de suerte el sábado, pero el domingo no pico nada. Así que, empezamos a usar una mosca llamada "hassle" en las zonas profundas. La probabilidad de capturar algún pez con la mosca hassle era escasa, ya que según nuestra experiencia las truchas la suelen ignoran. Después de un rato y a pesar del tiempo soleado, finalmente capturamos una trucha.

**Esta trucha fue atrapada con la mosca hassle y luego fue liberada**

## Modelo de Predicción en pesca

Después de volver a casa de nuestro viaje, formulamos un modelo informático de probabilidad para pescar basándonos en ciertas variables como el clima y el tipo de señuelo:

```
if (weather is cloudy) {
      probability = good;
}
if (it is evening) {
      increase probability;
}
if (tinsel is used) {
      increase probability;
} else if (hassle is used) {
      decrease probability;
}
if (fisherman is skilled) {
      increase probability;
}
```

Podemos probar este método en contra de lo que realmente sucedió, Pescamos en una soleada mañana de domingo, con una mosca hassle y con un pescador novato. A pesar de estos inconvenientes, finalmente capturamos un pez.

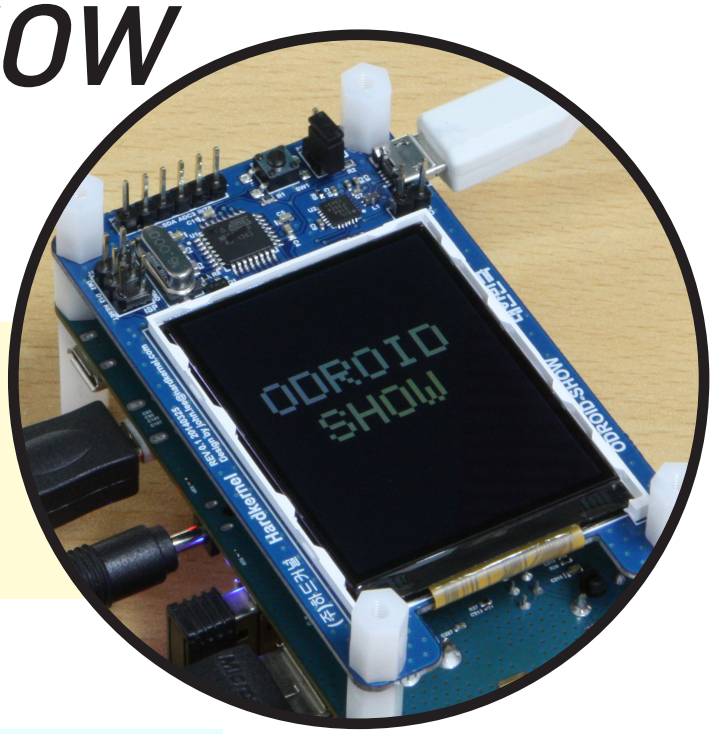
Si se quiere desarrollar un modelo mejor para capturar peces, podrías basarsete en el teorema de Bayes (`http://bit.ly/1nwkWIA`), que describe cómo los cálculos se pueden hacer con probabilidades condicionales. Un modelo más avanzado podría utilizar también algunos datos históricos del tiempo. Por ejemplo, saber si ha llovido en las últimas dos semanas o si ha hecho mucho calor. Estos datos de referencia podrían ser recogidos por una placa meteorológica con ODROID, ubicados en la zona de pesca y con los datos disponibles vía Internet.

**Un señuelo de mosca hassel tiene dos plumas del cuello de perdiz, y el anzuelo utilizado es colocado de forma que flote boca abajo**

**Señuelo Oropel – Una muestra de señuelo trucha oropel serpentina que imita a un pez pequeño**

# PROFUNDIZANDO (EN) EL ODROID-SHOW

## PARTE 2: HACIENDO CONEXIONES




En la primera parte de esta serie de artículos sobre el ODROID-SHOW, introduje los conceptos básicos del software que se puede ejecutar en éste, suficientes como para desarrollar tus propios programas. En este número, me centraré en el uso de los dos cabezales pin situados en la parte superior derecha de la placa para conectar algunos componentes sencillos al SHOW.

**Los dos cabezales pin de la parte superior derecha de la placa se puede utilizar para conectar varios componentes externos**

## Pines

Si apenas has empezado a usar Arduino, tanta información puede parecerte un poco desconcertante, así que voy a coger algunos párrafos y los voy a dividir en trozos más comprensibles.

Para empezar, las dos columnas más importantes son "Label", que es lo que está impreso en la placa y permite identificar el pin y la columna "Function", que identifica el uso principal para ese pin en concreto. Analizando la columna Function, deberías ser capaz de identificar los diferentes tipos de pines:

*** Pines de alimentación, que ofrece 3,54 voltios con un "alto" nivel lógico**
***Un pin de tierra, que es necesario para crear un circuito eléctrico y tiene un "bajo" nivel lógico**
*** Pines digitales, que pueden tener un estado "alto" o "bajo"**
*** Pines analógicos, conectados internamente a un convertidor analógico a digital que puede medir cualquier valor de tensión entre 0 (tierra) y 3,45 voltios (alto)**
*** Un pin de reinicio, que generalmente no se utilizada**

Cualquiera de los pines analógicos se puede programar para operar sólo como un pin digital si eso es lo que quieres, pero al contrario no es posible. Por lo general, los pines digitales se conectan a algún tipo de interruptores o botones, mientras que los pines analógicos están conectados a otras cosas como resistencias variables (potenciómetros) o resistencias dependientes de la luz. Ten en cuenta también que los pines sólo pueden funcionar en modo analógico cuando se configuran como entradas; cuando un pin es configurado como salida, sólo puede estar encendido o apagado pero no en un valor entre medias.

**Resumen de funciones de cada pin de ODROID-SHOW**

| Label | Function | Alt. Function | ATMega/Port | Pin Change Interrupt |
|---|---|---|---|---|
| 3v45 | Power | - | | |
| A5 | Analogue pin 5 | SCL (I2C clock) | 28/PC5 | 13 |
| A4 | Analogue pin 4 | SDA (I2C data) | 27/PC4 | 12 |
| A3 | Analogue pin 3 | - | 26/PC3 | 11 |
| D2 | Data pin 2 | Hardware Interrupt 0 | 4/PD2 | 18 |
| GND | Ground | | | |
| RESET | Bring low to reset/program chip | - | 1/PC6 | 5 |
| D11 | Digital Pin 11 | SPI MOSI / PWM / Output Compare Timer 2A | 17/PB3 | 3 |
| D12 | Digital Pin 12 | SPI MISO | 18/PB4 | 4 |
| D13 | Digital Pin 13 | SPI SCK | 19/PB5 | 5 |

## Funciones alternativas del Pin

Continuando, vemos que los pines pueden tener funciones alternativas, que se pueden dividir en grupos. Pueden ser utilizados para SPI, I2C, modulación con pulsos (PWM, que se usaa menudo para impulsar los motores o cambiar el brillo de los LEDs), interrupciones externas e interrupciones en el tiempo. Los dos primeros son protocolos diferentes que permiten al chip ATMega del SHOW comunicarse con los circuitos integrados más complejos (ICs), sensores, etc. Véase notas de "I2C y SPI" para una visión más completa.

No todas las funciones alternativas posibles están disponibles en ODROID-

SHOW por el hecho de que algunos de los pines también se utilizan para comunicarse con el hardware TFT. En concreto, los pines D11, D12 y D13 están conectados a la TFT, lo que significa que el pin D11 ya no puede ser usado para PWM o para poner en marcha interrupciones en el tiempo. Con el cableado y la programación correcta, se puede utilizar para comunicarse con otro dispositivo SPI aparte de la pantalla TFT.

La otra función alternativa es para las interrupciones externas que cubriré más adelante, cuando hable sobre el uso de las interrupciones en respuesta a las pulsaciones con botones.

La columna "ATMega / Port" muestra formas alternativas de tratar los pines. Si decides usar "avr-gcc" para escribir y compilar programas para SHOW en lugar del IDE "Arduino", necesitarás referirte a los pines usando la numeración de la documentación ATMega: "avr-gcc" no reconoce la numeración de los pines al estilo Arduino.

El valor "Port" es otro modo de referirse a los pines. El procesador ATMega agrupa pines en cuatro "ports" o " banks" separados e incluye métodos para leer o escribir cuantos pines se quiera en el mismo puerto y al mismo tiempo. También es posible configurar el manipulador de interrupciones para controlar los cambios del estado de los pines, pero (con la excepción de pin D2) esto sólo se puede hacer con un puerto completo en lugar de con pines individuales. Los puertos van de la "A" a la "D". Por ejemplo, el pin marcado como A5 en la placa tiene el nombre del puerto PC5, lo que significa que está en el puerto "C" y el bit 5 se usa para acceder a su valor.

Finalmente, cada pin tiene un número de interrupciones asociado que se muestra en la última columna. Cuando una interrupción "pin change " se activa, provoca una interrupción en el pin que va desde "high " a "low" o viceversa.

## Circuito de sensores

Dejando la teoría a un lado, podemos de hecho llegar a conectar algunos sensores y leer sus valores en el SHOW. Aunque son circuitos muy simples, a menudo es todo lo que se necesita para añadir algo de interactividad a un programa, en especial si deseas usar SHOW para una aplicación independiente.

Para ahorrar espacio, sólo voy a describir los puntos principales de cada circuito Arduino.

## Circuito pulsar botón

Se puede conectar un botón o interruptor en serie entre un pin digital y un pin 3v45. De esta forma, cuando se pulsa el botón o se cierra el interruptor, el pin registra un valor "high".

## Lectura de un potenciómetro

Los potenciómetros (o "pots") son una forma de resistencia variable. Pueden presentarse en formas circulares o lineales, como un botón de control de volumen o como un botón deslizador de una mesa de mezclas. Además pueden ser "lineal", donde la resistencia es proporcional al punto al se gira el "wiper", o "log" donde la resistencia es proporcional al logaritmo. Los pots lineales son, por lo general los más útiles.

## Lectura de joystick

La mayoría de los joystick, a excepción de los más viejos, son dispositivos analógicos, con un pot para el eje X y otro para el eje Y. Normalmente, tienen uno o más botones.

## Usar interrupciones para pulsar botones

Como resultaba simple conectar componentes ( resistencias variables, LEDs, interruptores, relés…) a un microcontrolador y los más complejos empezaron a ser más asequibles, la gente se dio cuenta de que necesitaban alguna forma estándar de interactuar con ellos. Los protocolos I2C y SPI fueron desarrollaron independientemente para abordar esta cuestión. La mayoría de los microcontroladores e incluso algunos procesadores potentes o SOC (System on a Chip) serán compatibles con uno o ambos.

Ambos protocolos tienen un bus compartido, sobre el cual los datos viajan hacia y desde el equipo periférico. Ambos utilizan una configuración maestro-esclavo, siendo el microcontrolador (MCU) el maestro que comunica a los dispositivos esclavos lo que deben hacer. También es posible conectar varios dispositivos diferentes al bus compartido, aunque varía el modo en el que es tratado cada dispositivo individual, es decir cómo el dispositivo sabe que el MCU le está hablando en un momento dado.

## SPI

Con SPI, Los diferentes dispositivos son tratados por una línea independiente "slave select" para cada dispositivo. Con el esclavo adecuado, el dispositivo conectado sabe que el MCU está hablando con él. Con I2C, por otro lado, cada dispositivo debe tener una dirección única que generalmente es un número de 7 bits. La mayoría de los dispositivos I2C pueden tener sus direcciones configuradas por medio de saltadores o puntos de estaño. El dispositivo maestro antepone a cada mensaje la dirección del dispositivo con el que desea hablar.

SPI tiene la ventaja de tener las más altas tasas de transferencia de datos con dispositivos esclavos, por lo que a menudo es usado en aplicaciones como pantallas TFT y módulos de tarjetas SD. También es capaz de realizar operaciones síncronas, lo que significa que los datos pueden ser transferidos en ambas direcciones a la vez sobre el MOSI ("Master Out, Slave In ") y pines MISO ("Master In, Slave Out "). La principal ventaja de I2C es que sólo necesita dos pines incluso si se cubre el número máximo de dispositivos. Esta característica hace que sea una opción muy común para una amplia variedad de sensores como la placa meteorológica para ODROID, dispositivos como los controladores de motor,

y placas de expansión E/S como la I/O shield para el U3.

También hay una gran cantidad de módulos disponibles para la plataforma Arduino que podrían ser conectados a los pines I2C o SPI del ODROID-SHOW. Lo único es que hay que asegurarse de que el dispositivo puede funcionar correctamente en el nivel lógico 3.45v que es el utilizado por SHOW. Incluso puede conectar dispositivos SPI y I2C al mismo tiempo, aunque tengas que dedicar un pin libre (A3 o D2) como una línea esclava para cada dispositivo SPI y evitar así conflictos con el controlador TFT.

La mayoría de los módulos de hardware diseñados para conectarse con Arduino o sistemas similares disponen de librerías y códigos demo disponibles para ello, por lo que usarlo en tu programa SHOW es tan simple como incluir la librería correcta y modificar la demo para que haga lo que necesites. Las librerías normalmente ocultan todos los detalles de los protocolos reales utilizados para la comunicación. Esto hace que sea muy fácil usar la mayoría de los módulos en tus propios proyectos, ya que dispones de una visión más amplia de lo que se supone que hacen y usar las librerías para gestionar las partes más monótonas.

## Seguridad eléctrica

Debido a las diminutas corrientes y tensiones usadas por ODROID-SHOW, no hay prácticamente ninguna posibilidad de que accidentalmente aparezca una descarga al conectar componentes pasivos a la placa. Sin embargo, los componentes electrónicos sensibles de SHOW es una cuestión a parte, ya que es muy fácil dañar la placa si conectas algo de forma incorrecta. Básicamente, hay tres formas de dañar los componentes electrónicos en el SHOW:

*** Crear una situación de sobretensión**
*** Crear una situación de sobre corriente (cortocircuito)**
*** Conectar una fuente de alimentación externa con una polaridad incorrecta**

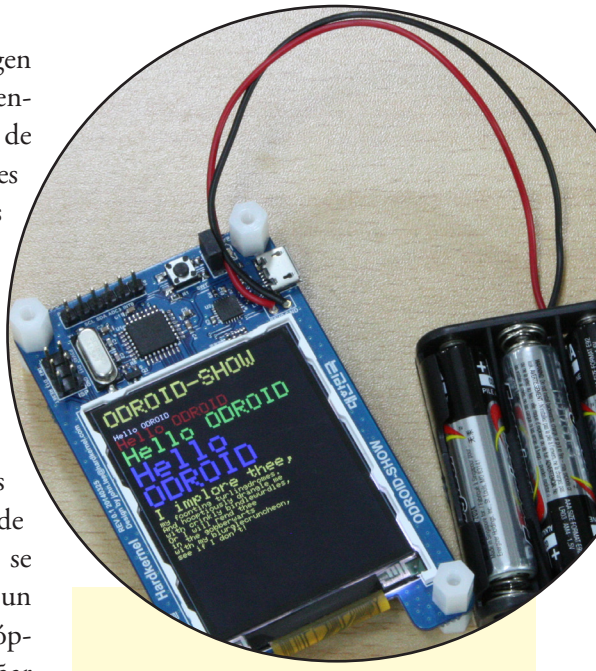
Las situaciones Sobretensión surgen al conectar un dispositivo con alimentación externa que suministra más de 3,45 voltios a cualquiera de los pines del SHOW. Esto incluye otras placas Arduino, que generalmente funcionan a 5V, o lo que se conoce como niveles "TTL" y algunos "módulos" con alimentación externa que pueden utilizar niveles de 5v. Estos dispositivos todavía pueden ser conectados siempre y cuando los pines 3.45V estén eléctricamente aislados de tensiones superiores, como cuando se usa un divisor de voltaje apropiado, un convertidor de nivel o un aislador óptico. La electricidad estática puede dañar algunos componentes, así que utiliza los métodos antiestáticos o de puesta a tierra adecuados cuando los manipules.

## Sobrecorriente

Las dos formas que puede causar una situación de exceso de corriente son la conexión de dispositivos que atraigan demasiada corriente desde los pines o por la formación de un corto-circuito. Los pines en un Arduino estándar (que funcionan a 5V) están pensados para un máximo de 40 mA en cualquier pin individual, con un consumo máximo de 200mA. El SHOW, sin embargo, funciona a 3.45V, por lo que la corriente máxima es menor. No he sido capaz de encontrar más datos sobre esta cuestión, así que para estar seguros recomendaría no superar los 20mA en cualquier pin o 100mA en conjunto. De hecho, desde que el SHOW también maneja una pantalla TFT que tiene una luz de fondo LED, puede ser que los 100mA sea el valor optimo. Algunos componentes, como los LED, podrían absorber demasiado corriente si se conectan directamente por lo que necesita una resistencia limitadora, mientras que otros, como los motores u otras cargas inductivas nunca deben ser activados directamente.

## Cortocircuitio

Un cortocircuito se forma cuando




**Panel LCD DE ODROID-SHOW con un pack de pilas adicionales**

se crea una conexión directa desde uno de tus pines +3.45v a tierra. Además del cortocircuito obvio, también es posible formar un corto con pines GPIO que se han configurado como SALIDA. Un pin de salida con una corriente alta de + 3.45v y uno con una corriente baja como GND tienen un riesgo de cortocircuito si los conectas a tierra o a un pin +3.45v respectivamente. Los pines configurados como entrada, por otro lado, son más seguros para conectar directamente cualquiera de tus pines positivos o toma tierra.

La solución es simple para los problemas de sobre corriente y cortocircuitos, asumiendo que no son descuidos y pasa por poner una resistencia con limite de corriente en serie con cualquier conexión que tenga un riesgo de absorber demasiada corriente o la posibilidad de darse un cortocircuito. Usando la ley de Ohm, un valor de 220 ohmios, por ejemplo, sería limitar la corriente a 3.45V / 220R = 15.7mA, que es seguro por debajo de la corriente máxima de 20 mA.

No obstante, es poco probable que cause daños, siempre y cuando identifiques los riesgos y compruebes el circuito antes de conectar nada..

# CLUSTER ODROID-U3 CON IO NODOS

## EL MEJOR Y MAS ECONOMICO SUPERORDENADOR CASERO

*Por Tynan Overstreet*

Siempre he soñado con tener mi propio superordenador. Por ello, me adentre en un campo donde abundan los potentes clústeres informáticos: el comercio algorítmico. En el mundo "algo", no es raro que las empresas pasen de unas cifras de seis a siete en sus clústeres con cada nodo nuevo suponiendo miles de dólares y consumiendo grandes cantidades de energía eléctrica.

Afortunadamente para los que no disponemos de demasiado dinero, el ODROID U3 basado en ARM nos permite desarrollar un clúster informático con la mínima parte del coste de una solución basada en el tradicional x86. Mi primer paso para alcanzar este objetivo, fue desarrollar un prototipo de clúster con 10 nodos U3, que actualmente están sometidos a una serie de pruebas de rendimiento frente a uno de mis nodos actuales basado en x86:

**-Hacer Backtesting con una nueva idea estrategia**
**-Filtrar la señal en tiempo real**
**-Crear caminos aleatorios**
**-El Desafío**

Este artículo incluye los resultados de la primera prueba: hacer backtesting con una nueva idea estrategia usando la denominada Response Surface Methodology (RSM), una potente técnica para estimar la formas y evaluar las funciones. En este caso, se trata de una híper superficie de 4 dimensiones. Cada punto de la superficie representa una posible configuración estratégica. Tres (3) de las dimensiones representan los parámetros

**Cada nodo del clúster de Tynan es un poderoso U3, cada uno con su propio ventilador de refrigeración**

configurables en la estrategia; pensar en ellos como botones o diales que controlan el comportamiento de la estrategia. La dimensión final representa la rentabilidad histórica de esa configuración.

La hiper-superficie puede ser imaginada como una piscina cúbica tridimensional, donde se mide la temperatura en cada punto, grabando también las posiciones x, y, z. Cuanto mayor es la temperatura, más dinero hace la estrategia. De este modo intentamos encontrar las más cálidas o más rentable regiones de la piscina. Debemos suponer que el agua caliente es algo bueno en este ejemplo, ¡y no el hecho de tener niños sin controlar en nuestra piscina!

Cada punto de medición toma una cantidad de tiempo no trivial para completarse. Debemos completar miles de mediciones para representar la superficie con precisión. En este ejemplo, estamos limitados por el tiempo que se necesita para evaluar realmente la lógica estrategia frente los datos históricos y medir el beneficio/pérdida resultante de cualquier operación hipotética.

### Los Competidores

Mi nodo actual consta de una CPU x86-64 con 8 núcleos a 4.2 GHz y ejectua Xubuntu con 8 GB de memoria, un SSD de 500 GG, GPU dual R9 270x para OpenCL (útil para generar caminos aleatorios), y una fuente de alimentación 850W ATX. Sin incluir la caja rack y el envío, el nodo cuesta alrededor de 1.150 dólares. Asimismo, después de comprar 10 U3 con los correspondientes módulos eMMC 8gb con Linux pre-cargado, fuentes de alimentación, ventiladores, y un switch, el coste de clúster ODROID fue de alrededor de 1250$, con menos de 100$ de gastos de envió. Yo prefiero tener algo de espacio en el disco local de cada nodo, pero si renuncias a los módulos eMMC y hacer un arranque de red, podrías ahorrar 250$ y construir un clúster ODROID con 10 nodos por alrededor de 900$, sin el envío.

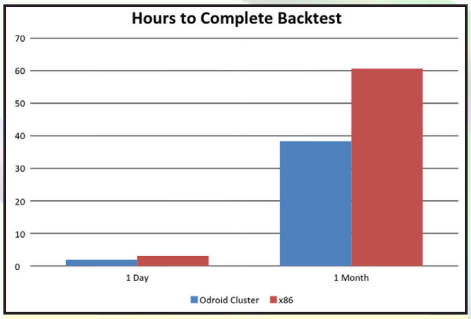
Además, cada U3 permite overclock

a 1.92 GHz, añadiendo la siguiente línea a /etc/rc.local justo antes del 0:

```
echo 1920000 > \
/sys/devices/system/cpu/cpu0/\
cpufreq/scaling_max_freq
```

## Resultados

Quise ver cómo actuaba el clúster ODROID con un código sin modificar. A excepción de una aplicación Python para coordinar el trabajo en el clúster, el código utilizado para backtesting fue el mismo que en mi nodo x86. Como se puede ver en la Figura, el clúster ODROID completa



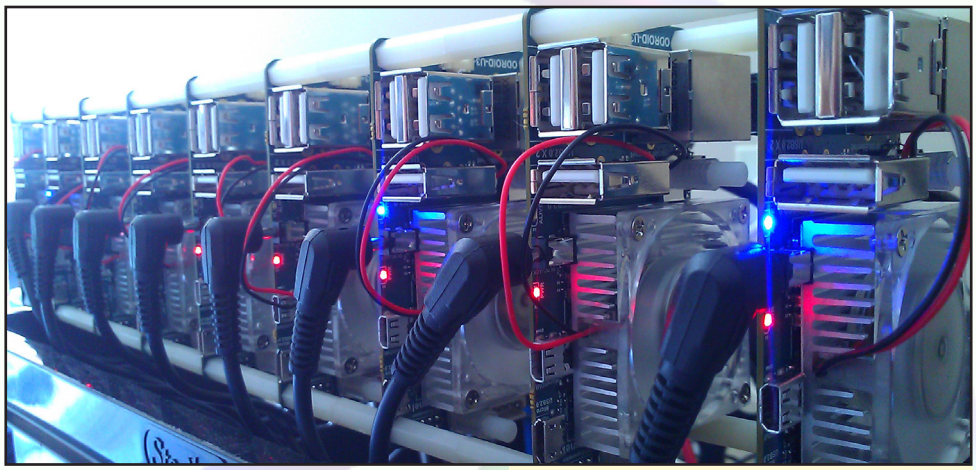

**Velocidad del clúster ODROID frente X86**

un backtest de 1 dia una hora más rápido que el nodo x86.

El Backtesting de un solo día no es muy útil, así que realicé un backtest de 20 días y grabé el tiempo transcurrido: los ODROIDs ofrecen una reducción de casi un 37% en el tiempo necesario para realizar la RSM, de una ejecución de más de 60 horas en el x86 a un día y medio en el clúster ODROID.

También hay un sinfín de beneficios menos evidentes en la ejecución de los ODROIDs frente al nodo x86. Por un lado, el clúster es prácticamente silencioso en comparación con el x86. Una vez que se enciende el LED azul apenas se nota que tienen un equipo encendido, y mucho menos 10. Por otra parte, el clúster U3 consume menos energía que el x86, por no mencionar la doble GPU.

El único inconveniente del clúster es la complejidad añadida al gestionar 10 nodos en lugar de un solo equipo. Así pues, estoy escribiendo una sencilla interfaz de administración de clúster que es específica para los ODROIDs y será liberada en www.ODROIDCluster.com cuando esté lista para su uso.



**La potencia total necesaria para este clúster U3 de 10 nodos es menos de 70 vatios**

Los siguientes son algunos errores comunes que sulenen aparecer al configurar el clúster por primera vez:

En primer lugar, tenía que regenerar la dirección MAC de cada ODROID para que en la red pudiese reconocer cada nodo como un dispositivo único. Esto se hizo eliminando el archivo / etc/ smsc_95xx_addr y reiniciando, lo que generó una nueva dirección MAC y me permitió asignar una dirección IP estática al nodo.

Luego, se retiraron los disipadores térmicos pasivos para añadir ventiladores de refrigeración activos, pero no es tan fácil como parece. Tuve que dejar que cada U3 reprodujera vídeos de YouTube durante 5-10 minutos antes de extraer los disipadores. ¡Este proceso permitió alguna exfoliación digital innecesaria!

Por último, la instalación de nfs-common no funcionó hasta que no actualice el kernel usando ODROID Utility. Para acceder a esta utilidad desde un U3, tuve que escribir sudo ODROID-utility.sh en mi línea de comando remota.

## Mejoras futuras

El código utilizado en esta prueba de velocidad era un código tal cual, sin modificaciones con el fin de analizar las diferencias de hardware de una forma rápida. Es posible mejorar el diseño del software aumentando, por ejemplo, el espacio de memoria no tenido en cuenta en esta prueba. El clúster ODROID tiene un total de 20 GB de memoria frente a los 8 GB del nodo x86, lo que significa que el software puede ser redi-señado para utilizar un disco duro menor, proporcionando incluso más velocidad.

## Seguimos Adelante

En la siguiente prueba comparamos los ODROIDs con el x86 en el procesamiento de señales en tiempo real procedente de datos de mercados. Los ordenadores aplicarán filtros bayesianos a los precios en tiempo real de futuros mercados de Estados Unidos. Hay posibilidad de mejorar significativamente el número de símbolos que el clúster puede procesar simultáneamente frente al nodo x86. Por último, probaré la capacidad de cada configuración de hardware para generar caminos aleatorios, que es necesario para fijar precios en tiempo real con las opciones de Monte Carlo.

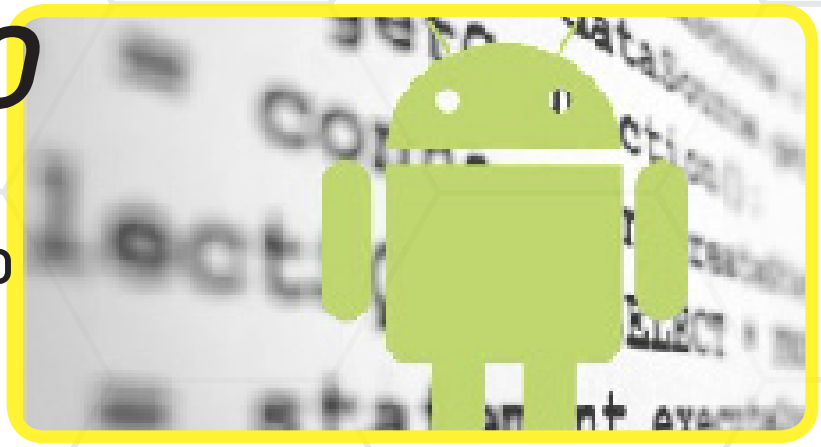
## Conclusión

El clúster ODROID se presenta como una alternativa muy atractiva frente a las tradicionales soluciones informáticas. En una prueba de velocidad usando la Response Surface Methodology, el clúster ODROID era un 37% más eficiente que mi nodo x86. Además, los beneficios adicionales como la disminución de ruido y de consumo de energía hacen que el U3 sea muy superior. Aunque gestionar un clúster conlleva una cierta complejidad administrativa, los beneficios superan con creces los inconvenientes. Para más información, visita http://www.ODROIDCluster.com.

# DESARROLLO ANDROID

## DENTRO DE LA APK ANDROID

*Por Nanik Tolaram*



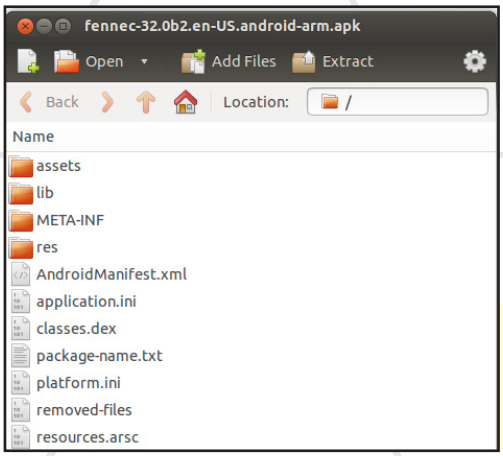
En anteriores números, presente de forma global las diferentes partes que componen los niveles internos de Android, y la forma en la que interactúan para crear el sistema operativo Android. En este artículo, vamos a explicar cómo desarrollar aplicaciones para Android. Lo mejor de la programación en Android es que puedes desarrollar y probar tus aplicaciones en cualquier dispositivo Android, y se ejecutarán (la mayoría de las veces) en el hardware de Android en su estado natural.

En el desarrollo convencional de Android, necesitas probar la aplicación con diferentes versiones de Android, pero esto no es necesario si estás utilizando el hardware ODROID. La familia ODROID de microordenadores es una gran plataforma sobre la que puedes probar tu aplicación, ya que es capaz de ejecutar diferentes versiones de Android. Sin embargo, ten en cuenta que Hardkernel no tiene soporte oficial para las versiones más antiguas como Honeycomb, pero publica imágenes para Ice Cream Sandwich 4.0, JellyBean 4.1 / 4.2 y 4.4 KitKat.

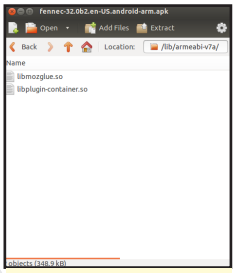
A lo largo de este artículo utilizare el navegador Fennec (Mozilla) de código abierto como ejemplo, que está disponible en http://mzl.la/1lxftFA.

## ¿Qué es un apk?

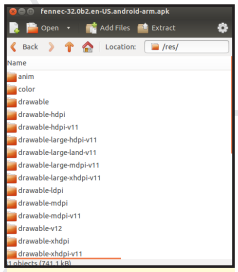
Un APK (Android Package File) es un único paquete comprimido que contiene todos los archivos necesarios de tu aplicación. Si con anterioridad has trabajado con Java, es similar a un archivo jar.



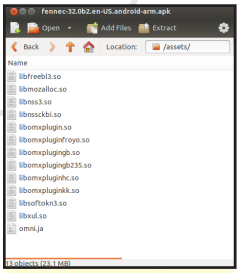

**Estructura de carpetas de un archivo APK**



**/lib**
**Contiene librerías nativas que dependen de la arquitectura**



**/res**
**Contiene recursos que son utilizados por la aplicación, tales como imágenes, texto y diseño**



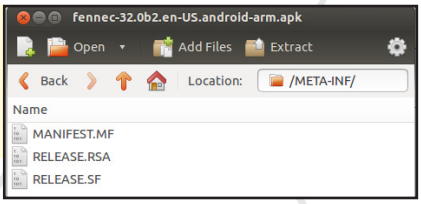
**/assets**
**Cualquier archivo que será utilizado por la aplicación puede ser colocado aquí. Normalmente los archivos incluidos en esta carpeta se leen como flujo de bytes y es adecuado para cierto contenido como imágenes pesadas, videos y otros formatos binarios**

La siguiente lista describe las diferentes carpetas y archivos:

**AndroidManifest.xml**
Este es el archivo de configuración XML que lee Android para localizar la estructura de la aplicación y otras propiedades.

**classes.dex**
Este archivo contiene la versión compilada del código de la aplicación.

**META-INF**
**Contiene MANIFEST.MF y el certificado de la aplicación**




**resources.arsc**
En general, todos los archivos APK que se ejecutan en Android tienen la misma estructura de carpetas. Este fichero es ligeramente diferente para los APK que se generan como parte de una imagen del sistema. Este contiene la versión binaria compilada de los recursos.

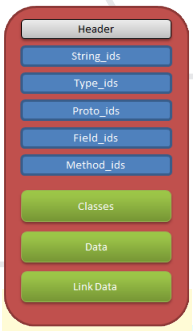
## Java vs NativO (C/C++)

Las Apps de Android se desarrollan normalmente utilizando lenguaje Java, pero hay muchas aplicaciones como los juegos, que se escriben en lenguaje nativo, que aún está disponible para Android. Las aplicaciones que se exportan a Android normalmente tienen algo de código Java, pero es usado como una capa por el código nativo. Un ejemplo de ello son los muchos juegos exportados, que se ejecutan en su versión original dentro de un "envoltorio" por Android.

Se recomienda escribir todas las aplicaciones de Android en lenguaje Java, ya que esto hará que sea más fácil exportalas entre las diferentes versiones de Android y mantiene la base del código al mínimo. Usar una aplicación nativa requiere volverla a compilar para cada versión de Android.

## El archivo .dex

Los Archivos APK de Android tienen un fichero llamado classes.dex que contiene el código de la aplicación en formato binario compilado. El formato de este archivo ha sido definido por Google y no es el mismo que el formato .class en Java. El archivo dex es más compacto que un archivo class normal y es necesario para que Android pueda ejecutarse en dispositivos más antiguos con almacenamiento limitado.




**Formato de un archivo dex**

Si escribes el siguiente comando, verás una copia de la información de classes.dex en el archivo fennec.txt. Si comparas el resultado con el formato de diseño .dex podrás analizar la información disponible en cada capa.

```
<sdk tool>/dx \
--dex --verbose-dump \
--dump-to=fennec.txt \
fennec-32.0b2.en-US.android-arm.apk
```

## Herramientas AOSP

Hay una serie de herramientas relacionadas con APK dentro de Android, como AAPT (Android Asset Packaging Tool), que es la principal herramienta para empaquetar aplicaciones. El siguiente diagrama de flujo muestra la secuencia de los pasos que se siguen para que una aplicación se ejecute en Android.

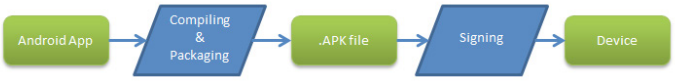



**Ilustración del proceso de empaquetado de APK**

El paso de compilación y empaquetado se realiza normalmente en un entorno de desarrollo interactivo (IDE), como Eclipse o Android Studio, ambos utilizan AAPT internamente.

## Dexdump

La herramienta Dexdump se utiliza para volcar el contenido del archivo classes.dex. Ejemplo del resultado:

```
Processing 'classes.dex'...
Opened 'classes.dex', DEX version '035'
Class #0            -
  Class descriptor  : 'Landroid/support/v4/accessibilityser-
vice/AccessibilityServiceInfoCompat$AccessibilityServiceInfo
VersionImpl;'
  Access flags      : 0x0600 (INTERFACE ABSTRACT)
  Superclass        : 'Ljava/lang/Object;'
  Interfaces        -
  Static fields     -
  Instance fields   -
  Direct methods    -
  Virtual methods   -
    #0              : (in Landroid/support/v4/accessibility-
service/AccessibilityServiceInfoCompat$AccessibilityServiceI
nfoVersionImpl;)
      name          : 'getCanRetrieveWindowContent'
      type          : '(Landroid/accessibilityservice/Acces-
sibilityServiceInfo;)Z'
      access        : 0x0401 (PUBLIC ABSTRACT)
      code          : (none)
…..
…..
…..
      access        : 0x0401 (PUBLIC ABSTRACT)
      code          : (none)
   source_file_idx    : 1418 (AccessibilityServiceInfoCompat.
java)
…..
…..
```

## Compatibilidad

Con cientos de dispositivos Android en el mundo, no es de extrañar que algunas aplicaciones funcionen perfectamente en algunos dispositivos y en otros aparezcan problemas. No hay una solución única para esta cuestión, ya que puede estar causado por código incompatible con el kernel o el código front-end de Android. En muchos casos, los vendedores modifican el código de Android para adaptarlo a lo que quieren conseguir, apareciendo así más complicaciones.

La mayoría de las veces, surgen problemas de compatibilidad con aplicaciones que interactúan con los periféricos de dispositivo, ya que pueden tener un comportamiento diferente cuando se ejecuta en varios hardwares. El problema no está en el propio código de Android, sino en el driver del dispositivo. Puesto que casi todos los drivers de hardware para Android son de código cerrado, no hay mucho que hacer para solventar el problema, excepto presentar un queja al vendedor.

El mejor estrategia para una aplicación es probarla en tantos dispositivos como te sea posible, o utilizar los servicios de terceros para ponenla a prueba en diferentes dispositivos a cambio de una tarifa. Otra técnica adoptada por muchos desarrolladores de software es liberar la aplicación permitiendo a los usuarios analizarlas en sus versiones beta, de este modo estás probando tu aplicación en una amplia variedad de dispositivos. Aunque no se recomienda este enfoque, a cambio puedes proporcionar a los usuarios actualizaciones gratuitas.

Para más información visita http://bit.ly/1A2T0l1, http://bit.ly/1uw6Xqc y http://bit.ly/1rLAfUK.

# CONOCIENDO A UN ODROIDIAN

## TOBIAS SCHAAF: UN NINJA LINUX Y UN AFICIONADO A ODROID

***Editado por Rob Roy***

**Tobias Schaaf, nuestro experto jugador montado en una avioneta**

*Por favor, háblanos un poco sobre ti.*

Mi nombre es Tobias Schaaf y soy de Alemania. Tengo 31 años y voy a cumplir 32 a finales de agosto. Soy un administrador de sistemas en una compañía de software que se centra en soluciones de software de medición inteligente, casas inteligentes y redes inteligentes. Todo lo relacionado con los ordenadores me ha interesado desde la primera infancia. Me encantan los juegos, leer libros, ver películas y animes, escuchar música y audiolibros. También me gusta nadar y por supuesto, todo lo relacionado con ODROID.

*¿Cómo fueron tus inicios con los ordenadores?*

Lo primero que me enganchó fue un viejo Atari 2600 que mis padres me regalaron en mi primera infancia. Más adelante, mi padre consiguió un Commodore 64 y después, un par de Amigas con unidades de disco duro. A los 14 años, tuve mi primer PC con Windows 95 que rompí en dos semanas, y que luego mi tío reconstruyo como un PC de arranque dual con DOS y Win95. Yo prefería mejor DOS que Windows. Desde entonces, me he asegurado de actualizar constantemente el hardware de mi equipo. Normalmente tengo 2-3 PCs funcionando activamente, además de algunos más viejos que están alrededor. Hoy en día, tengo una gran cantidad de ODROIDs que forman parte del hardware de mi PC.

**Tobías con un amigo preparando una barbacoa poco antes de su último viaje a los EE.UU.**

*¿Cuál es tu ODROID favorito?*

Hasta ahora, el X2 era mi favorito con un montón de puertos USB, un bonito diseño y de muy buena calidad. Tenía un interruptor para el eMMC y SD, con muchas funciones que me gustan bastante. Es por ello que fue mi primer ODROID, a pesar de que podría haber conseguido un U2 más barato. Pero al ver el nuevo XU3, que lo esperaba impacientemente y una vez que llego a mis manos uno de estos pequeñines, ¡se convirtió en mi nuevo favorito!

*Tu imagen GameStation Turbo es muy popular en los foros. ¿Qué otro software has desarrollado para ODROID?*

A veces, me pregunto si alguien más desarrolla "software" para ODROID. Sí, una versión de XBMC y un par de diferentes versiones del sistema operativo de Linux (Ubuntu, Debian, Arch, etc), que mucha gente estudia, aunque el software actual parece estar raramente exportado.

He exportado en su mayoría juegos y emuladores, que se pueden encontrar en la sección de Juegos y emuladores del foro. También he exportado algunos programas, como ClipGrab, sdl2 o ffmpeg. Además, mantengo mi propio kernel desarrollado desde Hardkernel como archivos .deb para instalar y actualizar.

Así mismo he creado archivos .deb para XBMC para instalar limpiamente y actualizar XBMC sin usar el script de actualización de @mdrjr, que simplemente copia los archivos de XBMC sobre una instalación ya existente.

La mayor parte de mi trabajo va directamente a la imagen GameStation Turbo, así que trabajo mayoritariamente con nuevos juegos y núcleos de actualización para Retroarch. Hasta donde yo sé, tengo la colección más grande de núcleos operativos que hay para ODROID y siempre trato de mejorarla. Trabajo en su estabilidad y añadiendo nuevas características al mismo tiempo. Recientemente, @AreaScout ha hecho muy buen trabajo ayudarme con estas tareas.

*¿Qué tipo de novedades en hardware te gustaría ver en futuras placas de Hardkernel?*

**1. Puerto (s) SATA**
**2. Puerto LAN 1000 Mbit**
**3. Dos puertos LAN 100 MBit para ser utilizado como router**
**4. Junto con el puerto SATA, una caja ODROID a medida que permita añadir unidades de disco duro a ODROID para construir un potente NAS**
**5. Un ODROID PC portátil como el OpenPandora.**
**6. U3 underclocked quizás a 1.2 GHz para reducir el consumo de energía, junto con un pack de baterías, un pequeño teclado y una simple pantalla. ¡Sería un buen proyecto!**
**7. Botones de encendido y reinicio. Funcionaron bastante bien en el X2**
**8. Mejores cajas que permiten el acceso a todas las partes del ODROID sin la necesidad de retirar la caja (por ejemplo, el eMMC, SD y puerto E/S) o una caja que permite un sistema de refrigeración más grande, tal vez un ventilador de 60 mm con un disipador de calor más grande.**

**A bordo del Nargoma, un barco museo al que le permitio a Tobias dirigir por un momento**

*¿Cuáles son sus planes de futuro para ODROID y tu Imagen GameStation Turbo?*

Tengo muchos proyectos activos actualmente. La imagen GameStation Turbo está en constante desarrollo y mejora. Tengo la intención de hacer más cambios en los núcleos y añadir algunos nuevos. Quiero cambiar el núcleo Amiga de Retroarch por fs-uae (otro emulador de Amiga) un emulador independiente, ya que viene con una versión de OpenGL ES que se ejecuta mucho más rápido que el núcleo Retroarch.

También estoy pensando incluir el emulador N64 en el que @AreaScout y yo estábamos trabajando. Todavía es mejorable. También detecte un problema con PPSSPP en Debian Wheezy, el sistema hace una pausa de vez en cuando en la emulación. Ocurre en la versión X86, así que se trata de un problema del propio Debian.

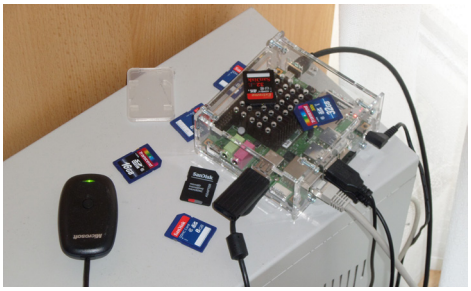
Además estoy considerando cambiarme a Ubuntu 12.04 o 14.04. Pero, tal vez no sea necesario. PPSSPP ha realizado algunas mejoras en los últimos meses, la versión más reciente soporta sistemas operativos ARMHF, mientras que las versiones anteriores no lo hacían. También han solucionado algunos problemas gráficos, así que tengo que comprobar cómo afecta todo esto a mi imagen.

Además de la GameStation Turbo, estoy trabajando en un proyecto para utilizar scripts para instalar diferentes tipos de servidores en un ODROID, por ejemplo, DNS, DHCP, Samba y Active Directory, que incluirá un menú fácil de utilizar.

También estoy haciendo un repositorio Debian "real", que te permite instalar y actualizar el software que he exportado a ODROID utilizando el comando aptget. También incluye la versión del kernel más reciente, por lo que podrás actualizar el kernel de ODROID simplemente escribiendo `apt-get dist-upgrade`.


Siempre estoy buscando los juegos más impresionantes para exportarlos a ODROID. Todavía tengo unas 50 fichas abiertas de juegos y programas en las que estoy pensando trabajar próximamente. ¡Hay mucho por hacer!

En la foto de abajo puedes observar mi querido X2 que sobre todo utilizo



para jugar ya que tengo un mando Xbox 360 conectado al mismo, pero como se puede ver por las tarjetas SD, también lo utilizo para hacer pruebas, junto con mis dos U3s.

También se muestra mi esclavo de desarrollo, un U3 conectado a un disco duro externo de 1TB, donde realizo mi magico software para mis proyectos.