



Introducing pi-web-agent, a Raspberry Pi Web App

A Web application allowing everyday users to control the Raspberry Pi.

VASILIS NICOLAOU, ANGELOS GEORGIADIS,
GEORGIOS CHAIREPETIS and ANDREAS GALAZIS

The pi-web-agent is a Web application that aims to give new users experience with the Raspberry Pi and potentially with any Linux distribution on any architecture. Raspberry Pi has introduced Linux to a lot of people, and we want to enhance their experience further. This project also demonstrates the extensibility capabilities of Linux.

What we provide enables you to install the pi-web-agent as soon as you have Raspbian (<http://www.raspbian.org>) installed. You'll simply be able to open your usual browser from your everyday machine and start interacting with your Pi.

The Product

The idea behind pi-web-agent is to support desktop environment functionality on browsers and provide extensions that behave similarly to mainstream desktop applications. If you have used GNOME or KDE, you will have noticed that each provides its own set of applications.

pi-web-agent is similar to Webmin (<http://www.webmin.com>), with the difference being that pi-web-agent targets everyday users. We want to give users a desktop experience through their browsers.

Main Features

The moment you launch



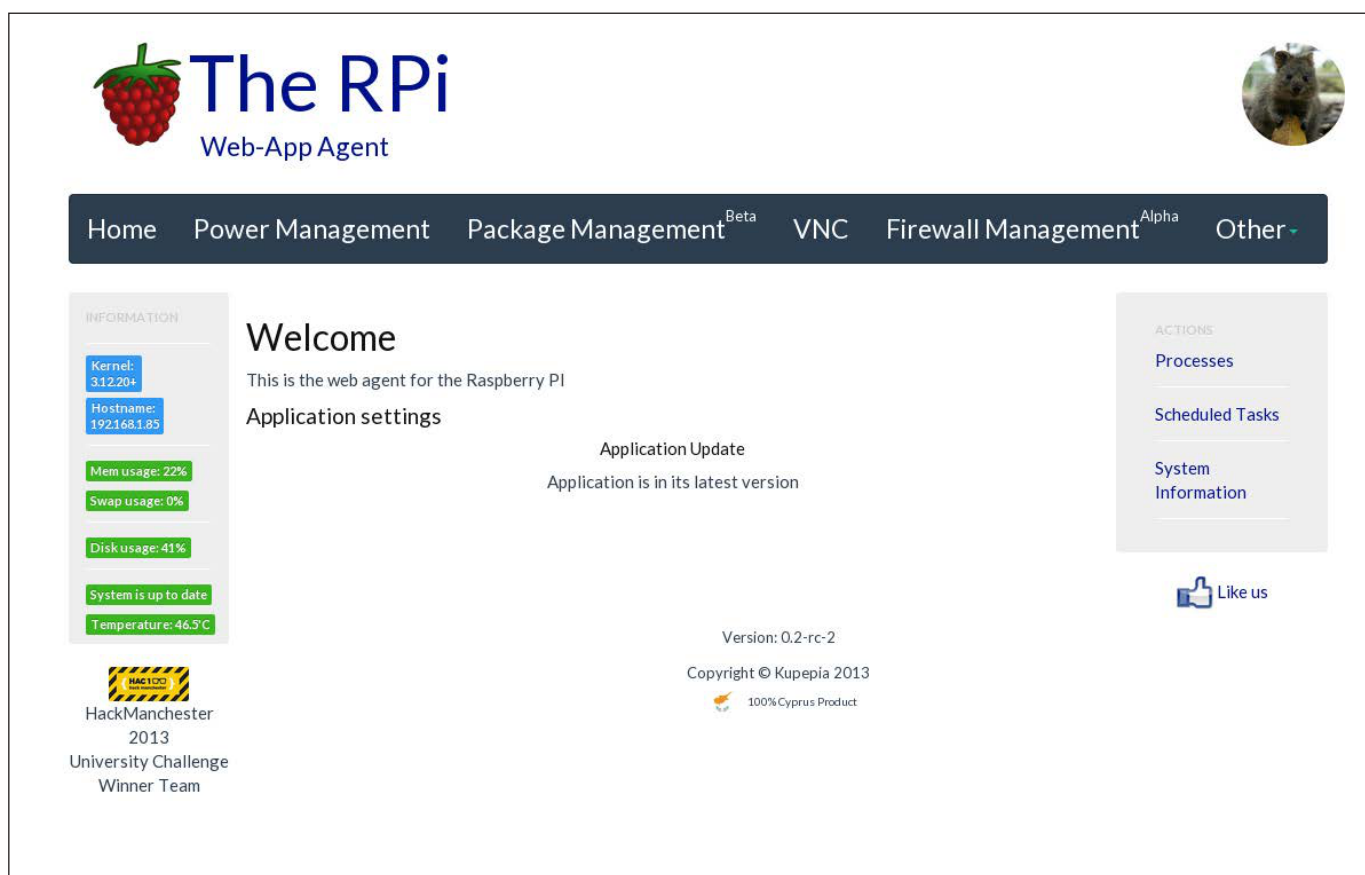


Figure 1. pi-web-agent Home Screen

pi-web-agent (after changing your password of course), you will realize that it already provides a lot of functionality despite its youth. On the left-hand side, you'll see some information concerning your Pi, such as its temperature, kernel version, update notifications, disk and memory usage.

The home screen (Figure 1) will check whether an update for the pi-web-agent is available. If it is, a button will appear that will download and install the new version of the application for you.

The navigation menu reveals the standard extensions that every desktop environment provides for users to interact with the underlying operating system, including power management for shutting down and restarting, and a package management section where we provide some of our favourite and useful applications with the capability to install them, enabling you to get started fast. One of the most important extensions is VNC, which provides the TightVNC server and gives you access to your real Pi



One of the most important extensions is VNC, which provides the TightVNC server and gives you access to your real Pi desktop with the Glavsoft TightVNC client Java applet.

desktop with the Glavsoft TightVNC client Java applet.

The firewall management, despite its early stage, enables you to avoid the fuss of many complicated options that iptables provides from the command line (until you become an

expert). You can set rules for various chains, block certain IP addresses or allow connections through different protocols (Figure 2).

Clicking the “Other” tab will reveal more extensions. The camera controller enables you to take snapshots and

The RPi Web-App Agent

Home Power Management Package Management VNC Firewall Management Other

Information:
 Kernel: 3.12.20+
 Hostname: 192168185
 Mem usage: 22%
 Swap usage: 0%
 Disk usage: 41%
 System is up to date
 Temperature: 46.5°C

fail2ban-ssh (Default Protocol: --)

protocol	target	otherinfo	destination	source	option
--	RETURN	--	ALL	ALL	--

OUTPUT (Default Protocol: ACCEPT)

protocol	target	otherinfo	destination	source	option
--	--	--	--	--	--

Actions:
 Processes
 Scheduled Tasks
 System Information

Like us
 Version: 0.2-rc-2
 Copyright © Kupepla 2013
 100% Cyprus Product

Figure 2. Firewall View When Clicking on a Certain Chain

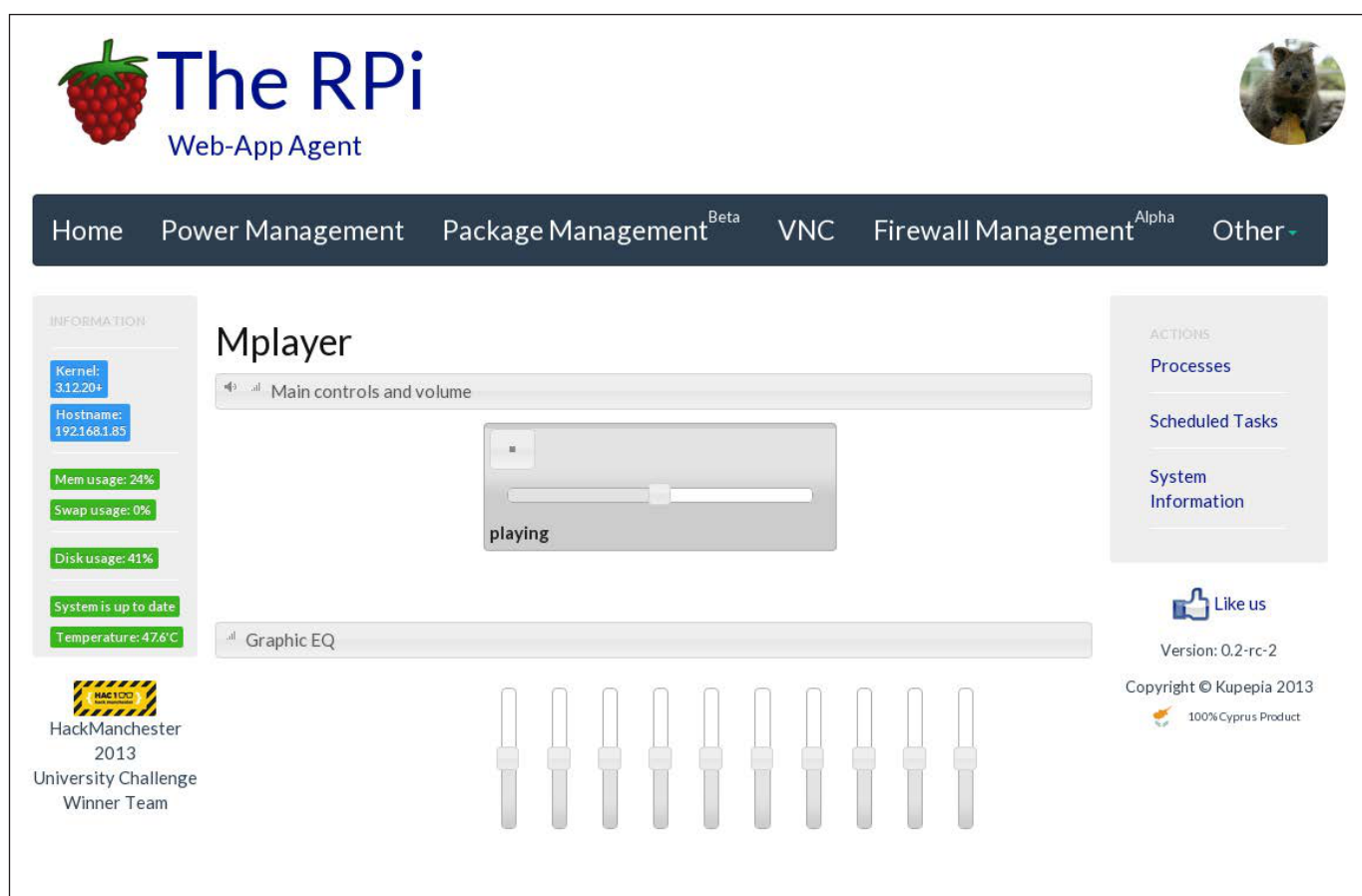


Figure 3. Media Player View When Listening to an Audio File or Streaming

even begin your own live stream! We also provide a media player (tagged as radio, since it started as such), where you can provide a URI of an audio file or a streaming radio channel. Your Raspberry Pi will start playing the audio, so get ready to attach your HD speakers!

You also can play an audio file straight from your Pi, but don't bother typing the URI in the text box. Find it through the file browser we provide. Although it's simple in functionality for now, it enables you to browse through

your files, and download or choose to play audio files with the pi-web-agent's media player extension. (Note: this functionality is available only from the development branch on GitHub, but it will be available in version 0.3.)

The media player extension is an Mplayer port that also enables you to control the sound with an equalizer (Figure 3).

If you want to be more hard-core and play with some wires and LEDs, we provide an extension for controlling the GPIO pins on your Pi



The RPi
Web-App Agent

Home Power Management Package Management^{Beta} VNC Firewall Management^{Alpha} Other

GPIO Manager
RPi.GPIO version: 0.5.5 RPi Board Revision: 2

INFORMATION

- Kernel: 3.12.20+
- Hostname: 1921681.85
- Mem usage: 22%
- Swap usage: 0%
- Disk usage: 41%
- System is up to date
- Temperature: 47.6°C

HackManchester
2013
University Challenge
Winner Team

DIRECTION	VALUE	LEFT	RIGHT	VALUE	DIRECTION
3V3	N/A	3V3	5V	N/A	5V
SDA	N/A	SDA	5v	N/A	5v
SCL	N/A	SCL	GND	N/A	GND
0 IN	OFF	GPIO7	TxD	N/A	TxD
0v	N/A	0v	RxD	N/A	RxD
0 IN	OFF	GPIO0	GPIO1	OFF	0 IN
0 IN	OFF	GPIO2	0v	N/A	0v
0 IN	OFF	GPIO3	GPIO4	OFF	0 IN
3v3	N/A	3v3	GPIO5	OFF	0 IN
MOSI	N/A	MOSI	0v	N/A	0v
MISO	N/A	MISO	GPIO6	OFF	0 IN
SCLK	N/A	SCLK	CE0	N/A	CE0
0v	N/A	0v	CE1	N/A	CE1

ACTIONS

- Processes
- Scheduled Tasks
- System Information

Like us
Version: 0.2-rc-2
Copyright © Kupepia 2013
100% Cyprus Product

Cleanup GPIO

Figure 4. The GPIO module gives you control over the Raspberry Pi GPIO pins.

(check out the interface in Figure 4). You also can check for updates or update your system and turn services on or off. There is more to come, so keep yourself posted via our Facebook page (<https://www.facebook.com/piwebagent>). We want you to forget you are using a Web browser and not bother clicking the VNC option.

How It Works

The pi-web-agent is served by

Apache (<http://www.apache.org>). Some claim that Apache is big and heavy, but we believe it has the most chances of incorporating cutting-edge technology. If you don't know what this means, check out mod_spdy with the SPDY protocol from Google (<https://code.google.com/p/mod-spdy>). We also can choose among a variety of modules that can increase pi-web-agent's potential. For those of you who want a lighter HTTP server, we are going to



release a pi-web-agent modification after the API is complete.

The core is written in Python and interacts via the Common Gateway Interface (CGI) to provide dynamic content. Our goal is also to provide an API, so almost every module is able to generate JSON data, making the application highly extensible both for us and for third-party developers. This will give us a lot of flexibility for future products we plan to deliver.

The styling currently is achieved with bootstrap (<http://getbootstrap.com>) using the Flatly (<http://bootswatch.com/flatly>) theme. User interaction mainly is achieved via JavaScript calls and rendering the document on the client side (most of the time). Our rule of thumb is to move as much processing from the Raspberry (server side) to the more powerful machine that runs the browser (client side).

Current Status

pi-web-agent is in a very early stage. It started in October 2013 at HackManchester (<http://www.hackmanchester.com>), winning the University challenge prize. The first version (0.1) was released on December 27, 2013, and the second release (0.2) followed in April 2014.

The current stage of development

has three phases:

- The development of version 0.3, which will introduce framework improvements. This also involves better extension management. As the number of provided extensions grows, we want users to choose what they want, and install and uninstall them at will. This will significantly reduce long installation times caused by dependencies (dependencies of the camera controller introduced 100MB of dependency packages).
- The development of the pi-web-agent for Android, which also includes the optimization of the API.
- The design and development of the pi-web-agent version 1.0. We want this to look like a real desktop environment, and we also want to achieve this in no more than a year.

pi-web-agent is open source, and it's easy for you to get involved—just fork our main git repository (<http://www.github.com/vaslabs/pi-web-agent>), and send us your changes through pull requests.

Using pi-web-agent

Imagine your Raspberry Pi has





just arrived and you have installed Raspbian on your SD card. Even if you don't have much experience with Linux and the command line, worry no more. You can connect to your Pi with SSH and install the pi-web-agent, which will help you in your first steps. While you become more experienced with the Pi and Linux, the pi-web-agent will grow with you, giving you more powerful capabilities and making your interaction with your Pi more enjoyable.

The most difficult task you'll face is the installation process, especially if you run a headless Debian distribution on your Raspberry Pi. You won't be able to avoid executing commands (until we release a Raspbian mod with the pi-web-agent included). You need to connect with your Pi via SSH. There are two ways to install the pi-web-agent, which are described below.

Installing through pstore

If you are using a Linux machine, it's easy. Just do:

```
ssh -X pi@raspberrypi
```

The -X will enable you to execute graphical applications. Provide your password to the prompt, and then launch the pstore by typing the

following and then pressing Enter:

```
pstore
```

When pstore opens, just register and search for pi-web-agent. Everything else is straightforward.

Installing via the Command Line

If you are not on a Linux machine, or if your distribution is headless, you still can install pi-web-agent easily. The following commands will fetch and install pi-web-agent:

```
wget https://github.com/vaslabs/\
    pi-web-agent/archive/0.2-rc-1.zip
unzip 0.2-rc-1.zip
cd pi-web-agent-0.2-rc-1
./install.sh
./run.sh
```

Troubleshooting

We've started a discussion on Reddit that covers a lot of troubleshooting, thanks to users' questions (http://www.reddit.com/r/raspberrypi/comments/249j4r/piwebagent_control_your_pi_from_the_ease_of_your). You can find guidelines on how to install under various circumstances and how to resolve problems that others already have faced. All the issues identified in this discussion have been resolved, but if you face a new one,





It is possible to extend the pi-web-agent by adding new Python modules.

just post a new comment.

Supported Platforms

The pi-web-agent framework is based on the micro-CernVM (Scientific Linux) appliance agent framework developed at CERN in summer 2013 (<https://github.com/cernvm/cernvm-appliance-agent>). We developed pi-web-agent based on that framework. We've modified it to work on Raspbian and cover the needs of the Raspberry Pi users. However, it is possible to use it on various Linux distributions with minor modifications concerning Apache configuration and replacing Raspberry Pi-specific modules, such as the Update and the GPIO.

We plan to release pi-web-agent version 1.0 for Raspbian, Pidora and Arch. Until then, the only officially supported platform is Raspbian.

Developing on pi-web-agent

It is possible to extend pi-web-agent by adding new Python modules. Upon creating a Python module, you'll find that the best way to work is to follow the structure below:

```
if 'MY_HOME' not in os.environ:
```

```
    os.environ['MY_HOME']=
        '/usr/libexec/pi-web-agent'
    sys.path.append(os.environ['MY_HOME']+
        '/etc/config')
    from framework import output

def main():
    output('Title', 'Hello my first module')

if __name__ == "__main__":
    main()
```

There are a lot of modules and methods you can use to get the most out of the framework. The most important is the output, which takes care of what's appearing on your browser. You can give two arguments, the title and the HTML, as the main content of your extension.

Framework Overview

The framework is composed of various Python modules and configuration files. The configuration files initially were in XML format, but they have been converted to JSON format, which reduced the codebase by a significant amount.

The core module is framework.py, placed in the /usr/libexec/pi-web-agent/etc/ config directory. This module





uses `view.py` and `menu.py`, which also use `HTMLPageGenerator.py` and `BlueprintDesigner.py` to construct the Web site skeleton. All the information about which modules are in use is in `config.cfg` in the same directory with the `framework.py`.

Adding Features

When you create your first module, you'll need to register it to the `config.cfg` file in order to be placed in the navigation menu. You'll also find that you can declare its version as Alpha or Beta. More options will be added soon, such as dependencies (next version 0.3) of a corresponding feature.

When a feature is added to the configuration file, the framework places it in the navigation menu with the URL provided in that file. There are two types of URL links: one for reloading the whole page and one for updating just the extension view (append `?type=js`). Since version 0.2, we started using the second format, and the first format is deprecated.

When clicking to select a feature, a JavaScript routine is triggered that loads the content of the extension in the appropriate area, where the user can interact with it. It is important to note that all user interface renderings will be performed on the client side

exclusively by version 1.0.

The Future

Reading through this article, you will have noticed that there are a lot of things pending and even more that can be improved. This is our goal: to develop a solid application, not only to satisfy users, but also to provide a good environment for other developers to extend or build on top of the pi-web-agent. That's why we have started multiple spin-off projects.

We have started a bunch of help projects in order to make life easier for us, users and third-party developers. We created a benchmark (<https://github.com/azardilis/testing-fw>) that gives us the loading times of each feature. We also started a plugin for the gedit text editor (the "official" text editor of our dev team) to automate the creation and deployment of pi-web-agent extensions.

Last but not least, we are developing pi-web-agent for Android. Not only have we started this application to increase user satisfaction, but it also is the driver for the pi-web-agent API, which will be given out officially, ready and documented, for third-party developers to build on. In addition, the API will be solely used for the creation of pi-web-agent version 1.0.





What Needs to Be Done

pi-web-agent is in an early stage, but it needs support from special tools to speed up the development process. Now that the framework has started to be more stable in terms of changes, we need to finish the gedit development plugin.

Next, we need to finish the pi-web-agent API very soon, and the pi-web-agent for Android will help shape a well-defined and documented API. Then, we will extend the framework to encapsulate the API modules both on client and server side. We plan to create the client-side framework using Dart (<https://www.dartlang.org>).

We also plan to start a new spin-off project that will be a Web site for hosting extensions for pi-web-agent, which users will be able to install via the Web application.

Our Team's Goals

Following the Android model, we want to build a platform to act as a desktop environment for the Raspberry Pi, which will be able to expand via extensions or Activities. A different Web site will act as a market and host those extensions. Developers will be able to register and publish their own extensions.

Get Involved

The project is small—as it should be

at this point. We want you, the users, to get involved before we start over-engineering things and making the pi-web-agent a big ugly piece of code that doesn't meet your needs.

There are a lot of ideas that need to be implemented. The pi-web-agent is already a product that is released in approximately a three-month cycle. You can become involved by sending us e-mail with recommendations, following us on GitHub, and forking and contributing to the repository. If you are new to any of these, don't worry; just send us your questions, and we'll get you started.

If visualizing a desktop environment in a Web browser is not so exciting for you, you can contribute to any of the spin-off projects that aim to boost the pi-web-agent development process. Here is a list of all the projects and their repositories:

- pi-web-agent:
<http://www.github.com/vaslabs/pi-web-agent>.
- Benchmark for testing the framework: <https://github.com/azardilis/testing-fw>.
- pi-web-agent for Android:
<https://github.com/vaslabs/pi-android-agent>.



- pi-web-agent gedit development plugin.

Conclusion

The pi-web-agent is a product that aims to replace the desktop environment with a Web-based alternative. HTML5 and CSS3

technology has made this possible. Along with the Linux extensibility, we chose to start with the Raspberry Pi, as it's the perfect platform for educational and experimental purposes. The Raspberry Pi also has the resource limitations we need, with the idea that if it runs fast on a

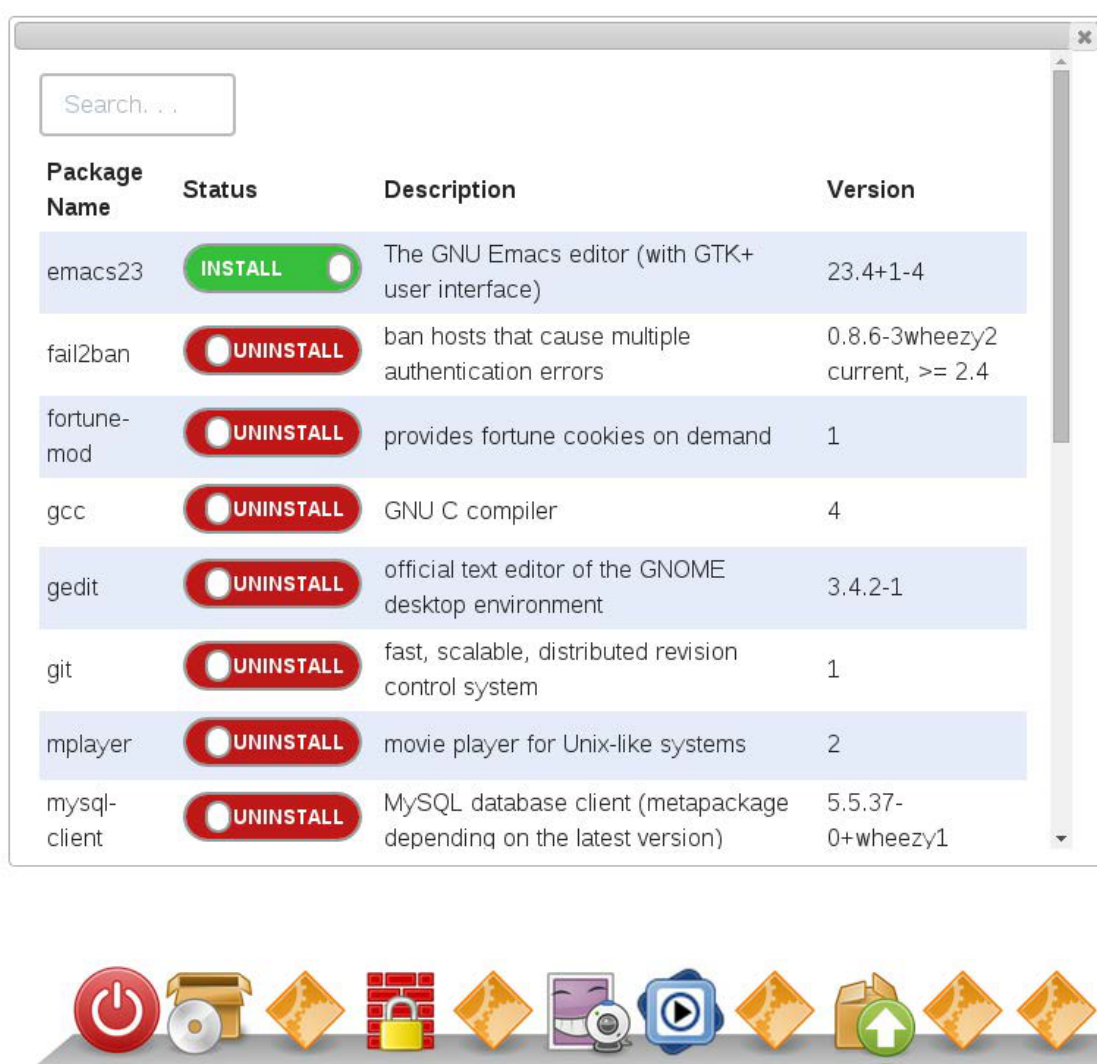


Figure 5. Sneak peek of the pi-web-agent version 1.0, simple window with content inside and dock as navigation menu.



Raspberry Pi, it will be super-fast on mainstream machines.

We already are developing a new design that brings the feel of a mainstream desktop environment. Figure 5, which demonstrates the dock navigation menu and the windowing system, provides a sneak peek of the new design.

As we mentioned already, we also need a lot of help. If you are a Python/CSS/HTML/JavaScript expert with some free time and a passion for open source, don't hesitate to contact us and join our team.

Acknowledgements

We want to give credit and a kind thank you to all the people that helped shape the pi-web-agent:

- Kyriacos Georgiou
- Maria Charalambous
- Argyris Zardylis
- Iliada Eleftheriou
- Theodoros Pertsas



Send comments or feedback via
<http://www.linuxjournal.com/contact>
or to ljeditor@linuxjournal.com.

Vasilis Nicolaou, pi-web-agent's founder, is maybe the biggest Linux lover on the team, at his job and maybe in other groups and places. He even knows what Linus Torvalds does daily better than Linus himself. Vasilis loves open source, Python, Java (don't tell that to Linus), Raspberry Pi and new technology in general, which (as Linus says) is what keeps him interested in programming. He graduated from the University of Manchester with an MEng degree in Software Engineering.

Angelos Georgiadis is a Java programmer who became a Python expert for the sake of pi-web-agent. He probably is the best chance of keeping pi-web-agent alive if Vasilis is hit by a bus tomorrow. He is the number-one suspect when something breaks in pi-web-agent and is probably responsible, since he has blown the repository quite a few times (17-ish).

Georgios Chairepetis is a young programming enthusiast, currently studying for an MSc in Software Engineering at the University of Manchester. He got involved with the rest of the pi-web-agent team initially by taking part in a hackathon contest and was lucky enough to win an award in the first competition he ever attended. He enjoys staying inside on Saturdays doing some programming with friends, but he also likes to go outside and enjoy the sunshine, maybe with some beer, when he has the chance.

Andreas Galazis has been a junior Web developer for six months. The fact that his favourite Linux application is Mplayer is somewhat misleading, as he spends most of his time coding rather than watching movies or listening to music, but when he does, he wants to do it the proper way. When he heard about pi-web-agent, he decided to join forces to develop an extension to demonstrate the power of his favourite media player.

