Brief instruction how to install birdhouse WPS on your local machine.

Here we describe the steps to install main components of the birdhouse WPS

All the components come from github repository:
https://github.com/bird-house

You can find detailed description of the services on techinal documentation web-site:
http://birdhouse.readthedocs.io/en/latest/

1. Requirements
2. Birdhouse components
3. Installation order
4. Phoenix web application to interact with Web Processing Services.
   Setup and first usage

1. Requirements

Birdhouse uses Anaconda Python distribution for most of the dependencies. If Anaconda is not already installed, it will be installed during the installation process. Anaconda has packages for Linux, MacOSX and Windows. But not all packages used by birdhouse are already available in the default package channel of Anaconda. The missing packages are supplied by birdhouse on Binstar. But we currently maintain only packages for Linux 64-bit and partly for MacOSX.

So the short answer to the requirements is: **you need a Linux 64-bit installation**.

Birdhouse is currently used on Ubuntu 14.04 and CentOS 6.x. It should also work on Debian, LinuxMint and Fedora.

Your installation user does not need any special permissions. All installed files will go into a birdhouse Anaconda environment in the home folder of the installation user.

For installation from the sources you need to have **git** installed on your machine.

(!) We also recommend to add the following line to your `~/.bashrc` file to make anaconda the Python found first than the system Python:

```
export PATH=$HOME/anaconda/bin:$PATH
```

2. Birdhouse components:

On the WPS client side we have:

- Phoenix: a Pyramid web application.
  http://pyramid-phoenix.readthedocs.io/en/latest/

- Birdy: a simple WPS command line tool.

  http://birdy.readthedocs.io/en/latest/

On the WPS server side we have:

- Malleefowl: provides base WPS processes to access data.
  http://malleefowl.readthedocs.io/en/latest/

- Flyingpigeon: provides WPS processes for the climate impact community.
  http://flyingpigeon.readthedocs.io/en/latest/

- Hummingbird: provides WPS processes for CDO and climate metadata checks.
  http://birdhouse-hummingbird.readthedocs.io/en/latest/

- BlackSwan: provides WPS processes focussing on extreme weather event assessments

- Emu: just some WPS processes for testing.

  http://emu.readthedocs.io/en/latest/

3. Installation order

Here we give an extraction from the installation procedures:
http://birdhouse.readthedocs.io/en/latest/installation.html

(!) Basically, you need to install **Malleefowl** and **Phoenix** and after you may install independently other services. To start with, try to install **Emu** service for testing, and **BlackSwan** for calculation of the weather regimes and analogs. During the run of blackswan processes, reanalysis data will be downloaded and cached – in this case, if you decide to use your own computer during the workshop, data will be already fetched. Other services may be installed any time later.

3.1 You need to create **birdhouse** folder in your home directory:

(terminal commands followed by the **$** sign)

```
$ cd
$ mkdir birdhouse
$ cd birdhouse
```

3.2 Install the very first backend service, **Malleefowl**:
http://malleefowl.readthedocs.io/en/latest/installation.html

check, that you are in birdhouse folder

```
$ cd ~/birdhouse
```

```
After,
```

```
$ git clone https://github.com/bird-house/malleefowl.git
$ cd malleefowl
$ make clean install
```

Now, after successful installation you need to start the services:

```
$ make start   # starts supervisor services
$ make status  # show supervisor status
```

If everything went OK you should see the status in terminal:
…
malleefowl ***RUNNING***
…

Now stop the service:

```
$ make stop
```

(!) Due to security reasons by default the local files, which are not from birdhouse cache are disabled. In order to allow using local files with file:/// prefix, one need to update custom.cfg file. For the start, you may skip this.

Open the file ~/birdhouse/malleefowl/custom.cfg
and add the following lines to the end of the file:

```
[pywps]
allowedinputpaths = /
```

Afterall the custom.cfg should looks like:

```
[buildout]

extends = buildout.cfg

[settings]
hostname = localhost
http-port = 8091
output-port = 8090
#log-level = WARN

## deployment options
#prefix =  /opt/birdhouse
#user = www-data
#etc-user = root

[pywps]
allowedinputpaths = /
```

To apply changes run

```
$ make update
```

(!) All make commands should be done in corresponding service folder, for malleefowl it is: ~/birdhouse/malleefowl/

You may change other options there, like hostname, ports, log-level etc.
Options from this file overtakes the base settings in ~/birdhouse/malleefowl/profiles/base.cfg
The important setting to be used later is http-port

(!) You always have to rerun make update after making changes in custom.cfg

(!) For every service independent conda environment will be created with the packages from environment.yml file (located in the service folder). And also for every service there will be it's own custom.cfg file.

Now, lets install GUI for WPS

3.3 Install **Phoenix** (web GUI):
http://pyramid-phoenix.readthedocs.io/en/latest/installation.html

```
$ cd ~/birdhouse
$ git clone https://github.com/bird-house/pyramid-phoenix.git
$ cd pyramid-phoenix

$ make clean install
```

By default the password for addministrator is '**qwerty**'
To change the password and get admin permissions run:

```
$ make passwd
$ make install
```

Your may also change the ESGF search api in custom.cfg
changing default one to

esgf-search-url = https://esgf-node.ipsl.upmc.fr/esg-search

(which is also not needed for the first time, but good to know)

and after run

```
$ make update
```

3.4 Installation of **birdy** – CMD tool

Birdy is a command line tool to work with Web Processing Services (WPS). It is using OWSLib from GeoPython project.

We will install it as anaconda package:

```
$ conda install -c birdhouse birdhouse-birdy
```

If it doesn't work – it means that you did not add $HOME/anaconda/bin to your ~/.bashrc file.

Instead, you may use

```
$ ~/anaconda/bin/conda install -c birdhouse birdhouse-birdy
```

Again, you can skip installation of the birdy – will work with it on workshop.

Now we will install main services that provide the processes for calulations.

The installation procedures are the same for all the services, so we will briefly describe it here:

3.5 Installation of **Hummingbird** – service provides processes for common tools in climate science like CDO etc.

```
$ cd ~/birdhouse
$ git clone https://github.com/bird-house/hummingbird.git
$ cd hummingbird
$ make clean install
```

3.6 Installation of **BlackSwan** - processes focussing on extreme weather event assessments

```
$ cd ~/birdhouse
$ git clone https://github.com/bird-house/blackswan.git
$ cd blackswan
$ make clean install
```

(!) Due to compilation dependency issue, it is absolutely nessessary to install specific hdf5 library under the main conda environmet. So, without activating any environments run (!) :

```
$ conda install hdf5=1.8.18=2
```

or

```
$ ~/anaconda/bin/conda install hdf5=1.8.18=2
```

3.7 Installation of **Flyingpigeon** - services for the climate impact community

```
$ cd ~/birdhouse
$ git clone https://github.com/bird-house/flyingpigeon.git
$ cd flyingpigeon
$ make clean install
```

3.8 Installation of **Emu** – service for WPS testing

```
$ cd ~/birdhouse
$ git clone https://github.com/bird-house/emu.git
$ cd emu
$ make clean install
```

3.9 This is optional, but you may allow local files also for flyingpigeon, hummingbird and blackswan. Just modify custom.cfg for each service the same way as for malleefowl by adding to the end of file:

[pywps]
allowedinputpaths = /

and after for each service run

```
$ make update
```

But this is not nesessary,

(!) After you finish the installation, you need to start the services. To do so, go to any service home folder, say ~/birdhouse/pyramid-phoenix/ and do make start:

```
$ cd ~/birdhouse/pyramid-phoenix/
$ make start
```

To stop the services, run:

```
$ make stop
```

To get the information about its status run make status. If everything works well, all the services should be in a running state:

```
$ make status

Supervisor status ...
bin/supervisorctl status
blackswan                        RUNNING   pid 31994, uptime 0:22:19
celery                           RUNNING   pid 31995, uptime 0:22:19
emu                              RUNNING   pid 31987, uptime 0:22:20
flyingpigeon                     RUNNING   pid 31988, uptime 0:22:19
hummingbird                      RUNNING   pid 32012, uptime 0:22:19
malleefowl                       RUNNING   pid 31997, uptime 0:22:19
mongodb                          RUNNING   pid 31986, uptime 0:22:20
nginx                            RUNNING   pid 31996, uptime 0:22:19
phoenix                          RUNNING   pid 31990, uptime 0:22:19
solr                             RUNNING   pid 32004, uptime 0:22:19
tomcat                           RUNNING   pid 31989, uptime 0:22:19
```

4. Phoenix web application to interact with Web Processing Services.

You may find the detailed description and tutorial of how-to use Phoenix GUI here:
http://pyramid-phoenix.readthedocs.io/en/latest/user_guide.html
http://pyramid-phoenix.readthedocs.io/en/latest/tutorial.html

(!) We will show all the steps and examples during the workshop but for practical work, if you want to use CMIP5 or CORDEX data, **you need to get ESGF OpenID**:

Get the openID for ESGF if you don't have one:
https://esgf-node.ipsl.upmc.fr/projects/esgf-ipsl/
https://esgf-node.ipsl.upmc.fr/user/add/?next=https://esgf-node.ipsl.upmc.fr/projects/esgf-ipsl/

Now, lets register **Blackswan** service in Phoenix and run one process (also follow the userguide from the link above):

```
4.1 check that everything is running (make status)

4.2 By default Phoenix runs on localhost. The HTTP port 8081 is redirected to
the HTTPS port 8443 so

4.3 Open https://localhost:8443 and approve the sertificates

4.4 Login as Phoenix administrator (with the password generated in section 3.3)

You may login with you ESGF OpenID, but in this case you will be in a guest
group. To add your ESGF user to administrators, you need to logout, login as
Phoenix administrator go to people section of the settings:
https://localhost:8443/people
And change the permissions for the user.
But for now - use Phoenix account. We will work with ESGF OpenIDs during the
training anyway.

4.5 Now, when you logged in, go to Settings:

Click to the Phoenix user in the upper right corner and select settings.
```

https://localhost:8443/settings

After select Services
https://localhost:8443/services

Register a new service:
https://localhost:8443/services/register

Fill the fields:

Service URL: http://localhost:8096/wps
Service Title: Blackswan
Service name: blackswan
Service type: Web Processing Service
Public access: do not select

And press: Register

BlackSwan is running on the 8096 port, for every service port is set in custom.cfg or in profiles/base.cfg

```
  http-port = 8096
```

You can register the Emu service in the same way, with the
http-port = 8094
title: Emu
name: emu

4.6 Now, you service is registered in Phoenix and you may select it in **Processes** or **Wizard** section of the upper menu. Processes used for processes with prescribed inputs (knows reanalysys, text input field etc.) If you need to select and search for the data, one need to use **Wizard** section.

So, if everything went well, go to **Processes** menu,
select **BlackSwan Service**
and the process: **Analogues of circulation (based on reanalyses data)**

Don’t change anything in the settings except set plotting of output to Yes, In the bottom of the page:

Plot: Yes

And press **Submit**

You will get to the monitoring page, where you can check the details of the job
https://localhost:8443/monitor

During the first run, the files of NCEP slp will be downloaded to :

~/birdhouse/var/lib/pywps/cache/blackswan

And later will be used without downloading.

Temporal calculations perfomed here:

~/birdhouse/var/lib/twitcher
~/birdhouse/var/lib/pywps/tmp

and removed after the successuful run

When the job is finished, you can find the results in the Output bookmark of the job. Check the maps with mean analogs and simulation (pdf) for example.

(!)
As resume, if you would like to try exercises on your local machine during the workshop, we recommend the following steps:

Install (without changing custom.cfg file for begining)

1. Malleefowl – section 3.2
2. Phoenix – section 3.3
   create the admin pass for phoenix user
3. BlackSwan – section 3.6
4. Emu – section 3.8

Do:
5. Get the ESGF OpenID
It will be needed to use IPSL virtual machine with access to ESGF data
Also check that you are able to download CMIP5 data (you need to register for CMIP5)

6. Register Blackswan end Emu in Phoenix
7. Run one process: Processes → Blackswan → Analogues of circulation (based on reanalyses data)
8. Check the output pdf and html

Thus you will have downloaded NCEP reanalysis files in WPS cache and installed main components of the birdhouse WPS.

After that during the training we will install command line tool, use owslib python client in jupyter notebooks with wps, try different processes from different services etc.