FATAL

Functional Analyses porTAL

Sebastien.Carrere@toulouse.inra.fr

FATAL automates the deployment of web portals required for analyzing sequence functions. It focuses on the analysis of transcript sequences (typically EST Clusters) and their corresponding peptides but can be used to analyze any sequence dataset.

Principle : from a set of sequence data and a configuration file, FATAL builds a set of shell scripts in order to :

1. **compute analyses, store and index results,**
2. **generate a web portal with standard bioinformatics tools (browser, search engine, blast and patscan servers).**

FATAL is written in PERL and uses scripts/libraries developped at LIPM (http://lipm-bioinfo.toulouse.inra.fr).
Data are stored into Lucene indices and BerkeleyDB files : no need to install any RDBMS.

FATAL has been used for different public proteome analyses (http://iant.toulouse.inra.fr/L.japonicus , http://iant.toulouse.inra.fr/G.max) and private several RNA-seq projects (http://www.heliagene.org/P.halstedii, http://www.heliagene.org/P.viticola).

FATAL has been written in the frame of BIOS project (http://bios.toulouse.inra.fr).

Quickstart section is built upon data taken from *Brachypodium distachyon* (see http://bios.toulouse.inra.fr).

For any question, please contact Sebastien.Carrere@toulouse.inra.fr .

Table of Contents

I. Pre-requisites

a. PERL modules

Many mandatory modules are included in FATAL tarball (directory lib/) but others must be installed on your system. All of these modules are available @CPAN or as debian package.

```
List::Util
Term::ANSIColor
IO::File
File::Copy
File::Basename
Storable
Proc::ProcessTable
CGI
Data::Dumper
GraphViz
HTML::Entities
JSON
DBI
Digest::MD5
Class::XML
POSIX
Lucene(*)
```

(*) : Lucene Perl module is built upon CLucene (C++ Lucene implementation). You have to install CLucene developpement files . This package is available for debian/ubuntu (libclucene-dev) and a tarball is freely available at: http://kent.dl.sourceforge.net/sourceforge/clucene/clucene-core-0.9.20.tar.gz

b. Binaries

These binaries (major part of them are Linux standards) have to be available in your sh/bash path :

```
sh
cp
rm
mkdir
cat
grep
tar
find
Mail
sort
gzip
base64
wget
echo
date
xsltproc
at
#usefull to create account but not mandatory
htpasswd
```

32 bits architectures : *if you are using a 32bits operating system, we provide some binaries for this kind of architectures ; just add the* ***`--32bits`*** *switch during the first installation step (FATAL_deploy.pl script).*

c. Useful tools

i) PARALOOP [strongly recommanded]
The program paraloop parallize the analyses on distributed ressources (SMP, SGE/PBS cluster, etc.) (http://lipm-bioinfo.toulouse.inra.fr/paraloop/)

If you have installed PARALOOP on your system, set the PARALOOP environment variable to the path where Paraloop is located.

```
#PARALOOP directory path
export PARALOOP=/path/to/paraloop-1.3/
```

Set the NCPUS variable to specify the number of cpus to use for parallelizing jobs.

```
#Depending of your system
export NCPUS=4
```

ii) InterPro/InterProScan

We use InterProScan (http://www.ebi.ac.uk/Tools/InterProScan/) tool to scan InterPro database. At the end, interproscan raw results are analyzed to automatically build a summary of the functional annotation based on protein domain content.

InterProScan requires a lot of cpu ressources ; if you don't have enough resources on the computer where FATAL is installed, you should compute InterPro analysis on another server. Many centers offer this kind of facilities, for example :

- Plate-Forme Bioinformatique du Genopole de Toulouse (http://bioinfo.genotoul.fr)
- Plateforme bioinformatique MIGALE (http://migale.jouy.inra.fr/)

d. Environment variables

If you have installed PARALOOP on your system, do not forget to set the PARALOOP environment variable to the path where Paraloop is located.

```
#PARALOOP directory path
export PARALOOP=/path/to/paraloop-1.3/
```

Set the NCPUS variable to specify the number of cpus to use to distribute jobs.

```
I.#Depending of your system
export NCPUS=4
```

II. Quickstart

This use case analyzes 100 EST clusters and 128 peptides. It tooks around 40minutes on a laptop (1 CPU and 4Go RAM); iprscan results are provided to speed up the process.

```
# Scripts have been tested with the bash shell

bash
rm FATAL.tgz
wget http://lipm-bioinfo.toulouse.inra.fr/download/FATAL/FATAL.tgz
tar xfz FATAL.tgz
cd FATAL

#Configuration

export FAT=$PWD
export WEBSERVER=http://server.domain.org
export HTTPROOT=/apache/path/to/FATAL
export EMAIL=your.mail@domain.org

#if you use paraloop set this ENV
#export PARALOOP=/path/to/paraloop/directory
#and tune this value to your system
#export NCPUS=2

$FAT/FATAL_deploy.pl --web_server=$WEBSERVER --http_root=$HTTPROOT --email=$EMAIL

#Brachypodium distachyon USE CASE: analyses and portal
cd $FAT/test/
tar xfz BRADI_data.tgz
export BRADI=$FAT/test/BRADI_data
cd  $FAT/site/db
ln -s $FAT/test/BRADI_data/tair .


cd $FAT
$FAT/FATAL_deploy.pl --species_cfg=$BRADI/BRADI.cfg \
                        --peptides=$BRADI/BRADI.peptides \
                        --clusters=$BRADI/BRADI.clusters \
                        --iprscan_result=$BRADI/BRADI.pep.iprscan \
                        --cluster_report=$BRADI/BRADI.report \
                        --reads=$BRADI/BRADI.reads \
                        --framed_pictures=$BRADI/BRADI.FrameDPictures.tgz \
                        --public_portal \
                        --main_id=accession \
                        --trust_db_format

cd $FAT/site/prj/BRADI/data/exec
(nohup sh process.sh > process.out) 2> process.err&
$FAT/FATAL_status.pl --species_cfg=$BRADI/BRADI.cfg
```

You can use `$FAT/FATAL_status.pl` *to follow the process execution .*

Then have a look to the website :

`http://server.domain.org/apache/path/to/FATAL/cgi/FATAL.cgi`

Search for example « `Brad000001*` » in the Quick Search box.

III. First install

Get and untar the FATAL tarball, then execute `FATAL_deploy.pl` with the following switches :

```
      --web_server=<url> [http://my.host.fr/]

      --http_root=<path> this is the "apache path" to the FATAL directory;
                    because it is a web interface , the rest of the world should access to this
directory using http protocol, and especially "cgi/FATAL.cgi" should be accessible
             NB1: do not forget to add ExecCGI option onto "cgi" directory
             NB2: try to access this CGI using a web browser (using web_server + http_root value)
from an external IP class (to avoid problems of security access)

       --email=<string> this FATAL instance administrator email
       --32bits : use this switch if you are using a 32bits system
```

```
bash
rm FATAL.tgz
wget http://lipm-bioinfo.toulouse.inra.fr/download/FATAL/FATAL.tgz
tar xfz FATAL.tgz
cd FATAL
export FAT=$PWD
$FAT/FATAL_deploy.pl --web_server=http://myhost.domain.org \
 --http_root=/FATAL/directory/apache/path \
 --email=someone@domain.org
```

Do not forget to use the `--32bits` switch if you are running FATAL on a 32bits system.

If any binary and/or perl module is missing, a **red message** will be written on your terminal.
Else, a **green message** with usefull information on how to configure your apache web server will be written with some CGI associated links to test.

A `.configure` file is written in `$FAT` directory to store all configuration mandatory variables. This file is used for the following calls of `FATAL_deploy.pl` and `FATAL_status.pl` scripts.

The following directories are created during FATAL installation :

`$FAT/web`
→ contains css files, pictures, logos, xml for portal layout, javascripts

`$FAT/bin`
→ contains iANT scripts (int subdirectory) and binaries provided with FATAL (ext subdirectory)

`$FAT/lib`
→ contains mandatory perl modules

`$FAT/site`
→ contains :

- all main configuration files (cfg subdirectory)
- web templates (`tpl`)
- temporary sessions directory (`tmp`)
- blast database repository (`db`), used for all portals deployed using this FATAL instance.
- `prj` : this directory will contain one subdirectory per project deployed : **if you untar an updated FATAL tarball, this directory won't be overwritten .**

IV. New Portal

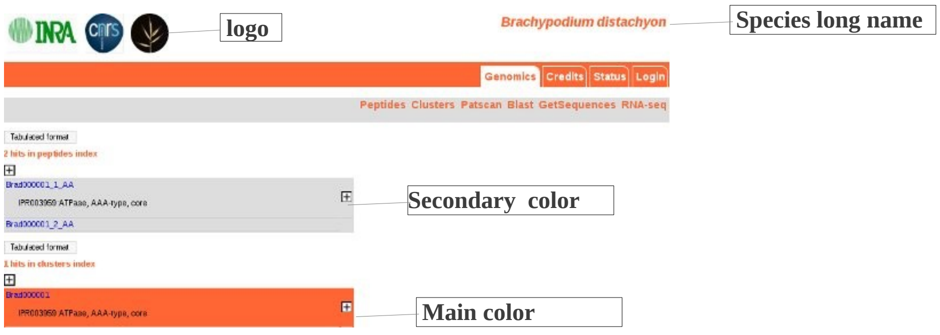
1. Species Configuration file

The first step to deploy a new portal is to create a configuration file describing the sequence data (species) and analyses to be performed.

This file is divided in 3 parts :

i) *Species generic information*

In this part you will describe the species shortname [**`species_code`**], species longname [**`species_longname`**], portal description [**`description`**], website look and feel [**`maincolor , secondarycolor`**], a logo for your species [**`png_url`** ou **`png_file`**] and an email address for the species portal manager.



ii) *Nucleic analyses*

In this part, you will describe analyses to perform onto clusters. They are all prefixed with a '**`clusters_`**' and suffixed by a unique number. Mandatory information for each analysis are :

- the method [**`method`**] (ncbi-blastx, ncbi-blastn,ncbi-tblastx, wu-blastx, wu-blastn,wu-tblastx)
- the target database [**`dbname`**] (in a FastiANT compliant format, i.e. with tags len=<sequence length> and acc=<unique accession> in fasta header (*))
- its description [**`definition`**]
- the maximum number of hits to report in final annotation sheet [**`maxdisplay`**]
- the specific parameters to use with the selected method [**`parameters`**]
- the number of processors to use with paraloop [**`paraloop_cpus`**] if it is different from `$NCPUS` (could be the case if you want to use a value greater than 1 for **WU-blast -cpus** or **NCBI-blast -a** parameters) .

iii) *Proteic analyses*

In this part, you will describe analyses to perform onto peptides. They are all prefixed with a '**`peptides_`**' and suffixed by a unique number. Mandatory information for each analysis are the same as the ones for nucleic analyses.

(*): We provide usefull fasta databases in FastiANT format at *`http://bios.toulouse.inra.fr/FATAL/site/db/index.html`*
To convert and index your own fasta files, you can use this script :

```
      $FAT/bin/int/iANT.aa_dna.FastaEntryDBConverter.pl --reformat --sequence /pathto/your/File.fasta
```

Here is a configuration file example :

```
#
# I. Species generic information
#

#Shortname (usually swiss-prot code)
species_code=BRADI
#Longname
species_longname=Brachypodium distachyon
#Portal description
description=Brachypodium distachyon functional analyses
```

```
#Website main color (tabs colors)
maincolor=#ff6633
#Website secondary color (submenus)
secondarycolor=#dddddd
#Species portal manager email
email=your.email@somewhere.org
#Species logo (display on welcome page and for self blast hits)
png_url=http://www.phytozome.net/images/home/orgs/bdi.png

#
# II. Nucleic analyses
#

# blast program (ncbi-?blast? or wu-?blast?)
clusters_method_01=ncbi-blastx
# Target database: this file should be available in $FAT/site/db if no full path is given
# Format : Fasta-iANT
clusters_dbname_01=pubmed
#Target database description
clusters_definition_01=UniProt subset with PMIDs (likely) not related to genome projects
#Max number of hits to display
clusters_maxdisplay_01=5
#Blast parameters
# WARNING : if you use Paraloop, do not forget to set -a 1 for NCBI blast or -cpus for WU-blast to one
to prevent the overload of your system

# The optimal settings depend on your target database size
clusters_parameters_01= -a 1 -e 0.000001 -b 100 -v 100 -K 200
# Cpus number to use for paraloop
clusters_paraloop_cpus_01=2
clusters_method_02=ncbi-blastx
clusters_dbname_02=tair
clusters_definition_02=A. thaliana proteome
clusters_maxdisplay_02=10
clusters_parameters_02= -a 1 -e 0.000001 -b 100 -v 100 -K 200

#
#III. Proteic analyses
#

peptides_method_01=ncbi-blastp
peptides_dbname_01=pubmed
peptides_definition_01=PubMed
peptides_maxdisplay_01=5
peptides_parameters_01= -a 2 -e 0.000001 -b 100 -v 100
peptides_paraloop_cpus_01=2
```

2. Input data

i) *Minimal set*

The minimal set of input data should be the species configuration file [**`--species_cfg`**] and the peptides multifasta file [**`--peptides`**].

ii) *Interpro analyses*

If you have performed Interpro analyses on your peptides dataset, you can provide the raw file to FATAL [**`--iprscan_result`**] . If you don't, FATAL will prepare a shell file to launch InterProscan.

Moreover, if you have already converted InterProsScan raw file into a description file (using our tool iprscan2annotation), you can provide it to FATAL [**`--iprscan_desc`**]. If you don't, FATAL will prepare a shell file to launch iprscan2annotation process.

iii) *Clusters*

Clusters multifasta file from which peptides have been predicted (using FrameDP for example) can be analysed and available through FATAL [**`--clusters`**]. Optionally, to store the information of cluster members , you have to provide a *multifasta like* file [**`--cluster_report`**]containing for each cluster the list of members in the following format :

```
>ClusterAccession1
```

```
ReadAccession1 ReadAccession32 ReadAccession89
>ClusterAccession2
ReadAccession61 ReadAccession2 ReadAccession457
```

iv) Reads

To complete cluster composition information, you can provide a list ofReads/EST fasta files (in FastiANT format) to get EST description in the cluster members links information [**--reads**] . These fasta files have to be compliant with the cluster report , i.e. Reads should have same ids

```
>ReadAccession1 acc=MyRead1Accession sp=MyRead1Species def=MyRead1 definition
AATCTGACATCGACTAGCATCAGCGACTACGACTACGACTAGCGACTACGACTACGACTACGATC
>ReadAccession61 acc=MyRead61Accession sp=MyRead61Species def=MyRead61 definition
AATCTGACATCGACTAGCATCAGTTTTTTTTTTCACGACTACGATC
```

v) FrameD pictures

If you have built your peptides using FrameDP program, you can provide a gzipped tarball [**--framed_pictures**] of clusters FrameD prediction pictures (these pictures are supposed to be named as `ClusterAccession1.<integer>.png`). These pictures will be available through the cluster annotation sheet.

3. Special switches

i) Authentication

If you want to deploy a public portal use the [**--public**] switch; however you can protect your portal later by setting the **public** parameter from **1 to 0** in the `$FAT/site/prj/$SPECIES/cfg/$SPECIES.cfg` file.
If you want to deploy a restricted access portal and you want to use an existing password file (from another portal), you can provide it using the command line parameter [**--password_file**]. This file has to be an Auth.pm-compliant password file (*) ; it will be copied into `$FAT/site/prj/SPECIES/etc/` directory.

(*) Auth.pm compliant format : You can use this script to produce such a file :

```
$FAT/bin/int/Authpasswd.pl
```

If you do not specify any of these 2 switches , by default your portal will be restricted to everybody (none account will be created)! You will have to create these accounts *a posteriori* using `$BIOS/bin/ext/Authpasswd.pl`

ii) Partial deployment

Sometimes, the server where you execute the analyses jobs is not the same as the one where the web portal would be hosted.
Use the [**--to_tar**] switch to create a tarball containing all analyses results and formatted databanks.
Then, transfer the file and rerun the script with the [**--from_tar**] switch to complete the installation (creation of the BerkeleyD and Lucene files and indices)

iii) BIOS related

If you want to add a BIOS (http://bios.toulouse.inra.fr) link on your cluster annotation sheet (assuming that your data are available in a BIOS repository with the same cluster accessions) use the [**--bios_repository=<your.bios.repository.uri>**] parameter.

4. Deploy your portal

```
#The FATAL root directory
export FAT=/path/to/FATAL/directory/
#PARALOOP directory path
export PARALOOP=/path/to/paraloop/directory/
#Number of cpus to use with PARALOOP (take care to the blast parameters)
export NCPUS=4
```

Launch the command according to your input data :

```
$FAT/FATAL_deploy.pl -species_cfg=</path/to/species.cfg> \
 --peptides=</path/to/peptides.fasta> \
 --iprscan_result=<path/to/iprscan.raw> \
 --clusters=</path/to/clusters.fasta> \
 --cluster_report=</path/to/clusters.members> \
 --reads="/path/to/reads1.fasta,/path/to/reads2.fasta,/path/to/readX.fasta"
```

The script will perform controls, create project directories and generate shell scripts in

```
        $FAT/site/prj/$SPECIES/data/cmd/*.sh
```

- these files contain command lines for each analysis step
- these files are prefixed according to their queueing order to prevent from inconsistencies :

file format conversion	00.fasta2xml.sh
(1)interpro analyses	01.interpro.sh
peptides automatic annotation based on interpro	02.autoannot_peptides.sh
annotated fasta creation per peptide	03.fasta_annotated_peptides.sh
annotated peptides databank creation	04.annotated_db_peptides.sh
clusters automatic annotation based on peptides annotation	05.autoannot_clusters.sh
annotated fasta creation per cluster	06.fasta_annotated_clusters.sh
annotated clusters databank creation	07.annotated_db_clusters.sh
clusters/peptides db slicing (500seq/file) to speed up blast	08.split_fastadb.sh
cluster members insertion (if required)	12.addclustermembers.sh
wu-blastn clusters vs. clusters	20.00.clusters.wu-blastn.<SPECIES>Clusters.sh
clusters blast analyses (as many as describe in species.cfg)	20.<INDEX>.clusters.<METHOD>.<DBNAME>.sh
clusters annotation xml sheet creation	25.clusters_create_sheet.sh
(2)Lucene sub-index creation (one per 5000 clusters)	26.01.clusters_idx.sh
(2)Lucene sub-index merging	26.02.clusters_idx_merge.sh
(2)Raw results storage in BerkeleyDB file	27.clusters_bdbh.sh
ncbi-blastp peptides vs. peptides	30.00.peptides.ncbi-blastp.<SPECIES>Peptides.sh
peptides blast analyses (as many as describe in species.cfg)	30.<INDEX>.peptides.<METHOD>.<DBNAME>.sh
peptides annotation xml sheet creation	35.peptides_create_sheet.sh
(2)Lucene sub-index creation (one per 5000 clusters)	36.01.clusters_idx.sh
(2)Lucene sub-index merging	36.02.clusters_idx_merge.sh
(2)Raw results storage in BerkeleyDB file	37.clusters_bdbh.sh

(1): this file should be empty if you provide iprscan raw results AND formatted description file associated; if you provide only iprscan raw results this file should contains the iprscan2desc command line.

The main shell script is created there :

```
$FAT/site/prj/$SPECIES/data/exec/process.sh
```

- this file contains all the analyses shell scripts call in the sequential order
- it also contains parallelization instructions

To launch all analyses in one run, just type the following commands :

```
bash
cd $FAT/site/prj/$SPECIES/data/exec/
(nohup sh process.sh > process.out ) 2>  process.err&
```

For each analysis step, the shell goes into the corresponding directory (`$FAT/site/prj/$SPECIES/data/exec/<INDEX>`) . There, a starting file is written, then PARALOOP files when this analysis is parallelized. When the jobs are finished, an ending file is created.
You can follow the whole execution process using the command :

```
$FAT/FATAL_status.pl -species_cfg=</path/to/species.cfg>
```

You can also have access to this status using the web interface of your species portal (Status tab) :

At the end of the process, if everything is okay (do some tests on the web site), you may backup and remove the `$FAT/site/prj/$SPECIES/data/xml` directory. It is no more used by the portal. The data should be stored, compressed, in BerkeleyDB files in `$FAT/site/prj/$SPECIES/data/bdbh` and indexed by lucene (`$FAT/site/prj/$SPECIES/data/indices)` .

V. Customizing the portal

1. List of Blast/Patscan target databses

Edit the files :

- `$FAT/site/prj/$SPECIES/cfg/$SPECIES_blast.cfg`
- `$FAT/site/prj/$SPECIES/cfg/$SPECIES_patscan.cfg`

The index number 00, 10 , and 20 are allocated to species Reads, Clusters and Peptides.

2. List of fields to be searchable in the browser

Edit the file :

- `$FAT/site/prj/$SPECIES/cfg/$SPECIES.idx.cfg`

cf. documentation de EZLucene http://lipm-bioinfo.toulouse.inra.fr/download/EZLucene/

Searchable fields throught the web form (Browse sub menu) are the ones indexed as Text or Unstored.The ones treaten as Unindexed are just stored in Lucene index.

```
#Fields indexed and stored in lucene index
#syntax: field:<alias>=Text.<xpath query>
field:accession=Text./entry/@accession
field:product=Text./entry/definition[@id='product'][@ontology="INTERPRO"]

#Fields indexed but not stored in lucene index
# => can be queried but can not be retrieved
#syntax field:<alias>=UnStored.<xpath query>
field:id=UnStored./entry/@id

#Fields not indexed but stored (gzipped and b64 converted) in lucene index
# => can be retrieved but can not be queried
#syntax field:<alias>=UnIndexed.<xpath query>
# <xpath query> = "file" means that the raw file is stored
field:b64_gz_xml=UnIndexed.file

#Additional parameters
field:hit_bank=Text./entry/entry_features/feature/database/@name
#Field description (for help table)
hit_bank_description=Search for a cluster which have a blast hit against this database
#Field datatype (for help table)
hit_bank_datatype=alpha-numeric
#Field list of values for autocompletion
hit_bank_complete=BRADIClusters,BRADIPeptides,tair,interpro
```

3. Look and feel

Edit the files :

- `$FAT/site/prj/$SPECIES/cfg/$SPECIES.cfg` in order to change `color_summary_lucene_clusters` and `color_summary_lucene_peptides`
- `$FAT/web/css/$SPECIES.css`

4. Add a menu/sub-menu

File to edit :$FAT/web/xml/$SPECIES.xml
This file contains the website layout (menus, sub-menus) and cgi functions attached.
cf. WebBuilder.pm documentation (perldoc $FAT/lib/WebBuilder.pm)

i) first, you have to add this tab in the xml file `$FAT/web/xml/$SPECIES.xml`

ii) then, you have to write the attached function which will be called ;add this function/procedure in the file `$FAT/lib/$SPECIES_cgi.pl`

This file won't be overwritten if you update your FATAL instance neither if you deploy once again the same species portal (with updated data for example).However, xml file `$FAT/web/xml/$SPECIES.xml` will be overwritten, but a copy of your previous xml file will be done `$FAT/web/xml/$SPECIES.xml.bkp.<YYYYMMDD>.<timestamp>`

5. **Add a template for a new blast target database result display**

Create in `$FAT/site/tpl/` a file named:

- **feature_***BlastMethodDatabaseName***.xhtml** : if you have specific template for a database AND a blast method
- **feature_***DatabaseName***.xhtml** : if you have specific template for a target database

`DatabaseName` correspond to the target database filename without file extension.

For example:

- if you want to use a specific template to display ncbi-blastp or wu-blastp results against `/path/to/nr.fasta` database, create a file named **feature_blastpnr.xhtml** (wu or ncbi are ignored)
- if you don't care about which method was used to blast against `/path/to/nr.fasta` database, just create a template named **feature_blastpnr.xhtml**

Template processor looks first for :

- first `$FAT/site/tpl/`**feature_***BlastMethodDatabaseName***.xhtml**
- then `$FAT/site/tpl/`**feature_***DatabaseName***.xhtml**.
- and finally templates declared in `$FAT/site/cfg/iANT.tpl.cfg`

To create those templates, use existing ones to help you. These these files are used by perl module `AnotherTemplate.pm` (`perldoc $FAT/lib/AnotherTemplate.pm`) with objects from `iANT` library (`entry`,`hit`,etc...).

If you want to also add a logo for this database, you can add it in the `$FAT/web/img` directory and use one of the method
`#PERLLIB|iANT/Utils.pl|GetWebPngUrl("`*ImageName*`")#`
(will look for a file named `$FAT/web/img/`*ImageName*`.png`) or
`#PERLLIB|iANT/Utils.pl|GetWebJpgUrl("`*ImageName*`")#`
(will look for a file named `$FAT/web/img/`*ImageName*`.jpg`)

You can also use hypertext link image in your template to get remote picture!

6. **Modify credits page**

Edit the file `$FAT/site/prj/$SPECIES/tpl/credits.xhtml`