6° Encuentro Linux

PEAR

Usando la infraestructura de PEAR para distribuir programas

6° Encuentro Linux
20-22 de Octubre del 2005
Iquique, Chile

Jesús M. Castagnetto, Ph.D.
<jmcastagnetto@php.net>

http://www.castagnetto.org/

6° Encuentro Linux

Agenda

- ¿Qué es PEAR?
- La comunidad de PEAR.
- Los estándares de código y de documentación.
- Componentes de un paquete.
- Como crear un paquete para distribución usando PEAR.
- Crear un canal para distribuir paquetes.

PEAR

PHP
Extensions and
Applications
Repository

... y sí, el sitio es todo verde

¿Qué es PEAR?

- Un repositorio de código reusable, sólido, bien mantenido, y que ha sido revisado por muchos.
- Una infraestructura para manejo y distribución modular de paquetes.
- Una comunidad de programadores con un mecanismo de interacción técnico y social definidos.

Metas de PEAR

- Proveer paquetes generales e interoperables al usuario.
- Requerir alta calidad en el código, y mantener un estilo consistente y claro de programación.
- Asegurar una mejora continua y un mantenimiento sostenido de los paquetes proveidos.
- Crear un ambiente para el libre intercambio de ideas y experiencias.

6° Encuentro Linux

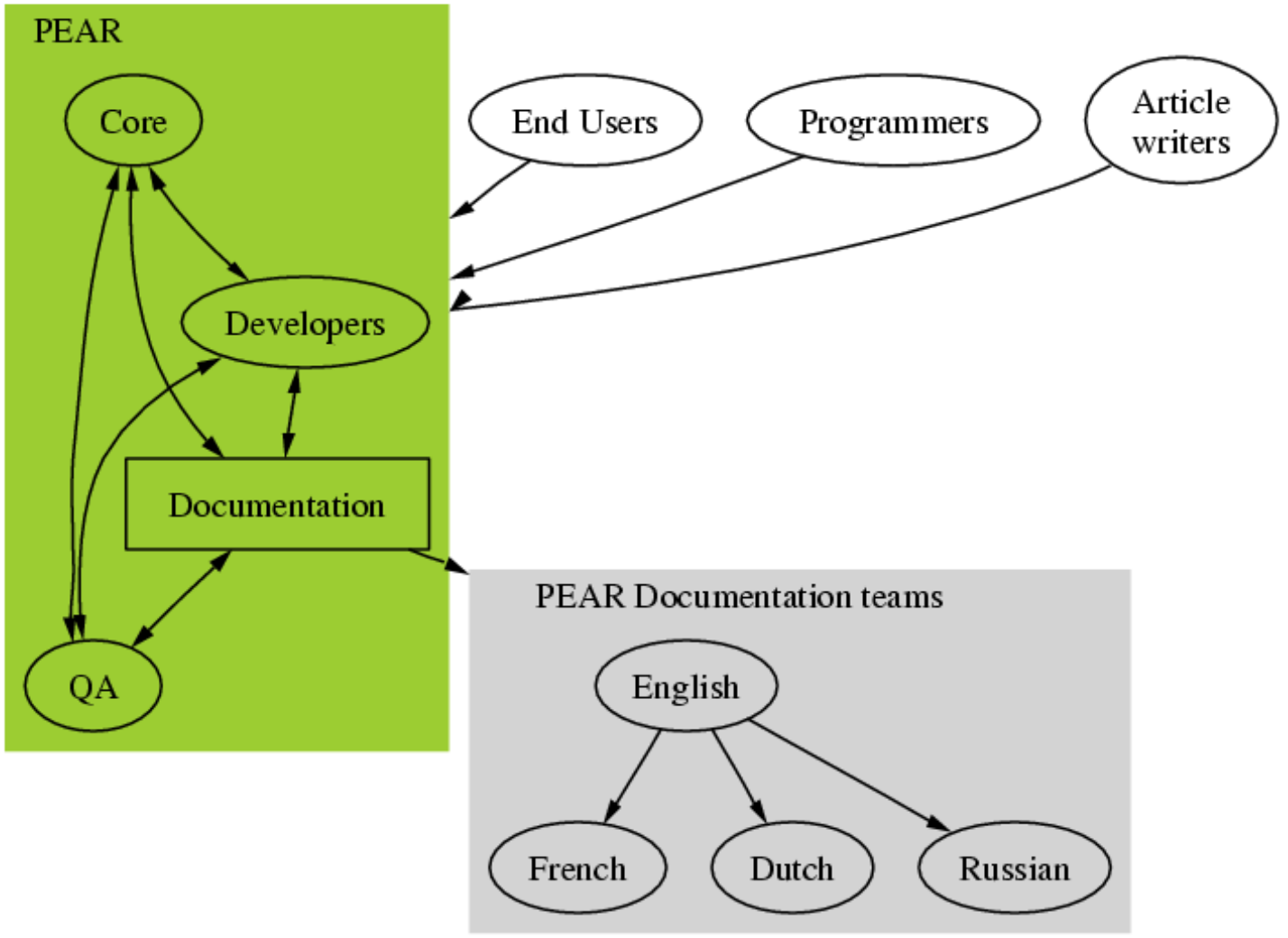
El Ecosistema

Ecosistema: El complejo de una comunidad de organismos, medio ambiente, e interrelaciones, visto como una unidad ecológica funcional.

(*Traducido de* '*ecosystem*' *del* "*Merriam-Webster Dictionary*")

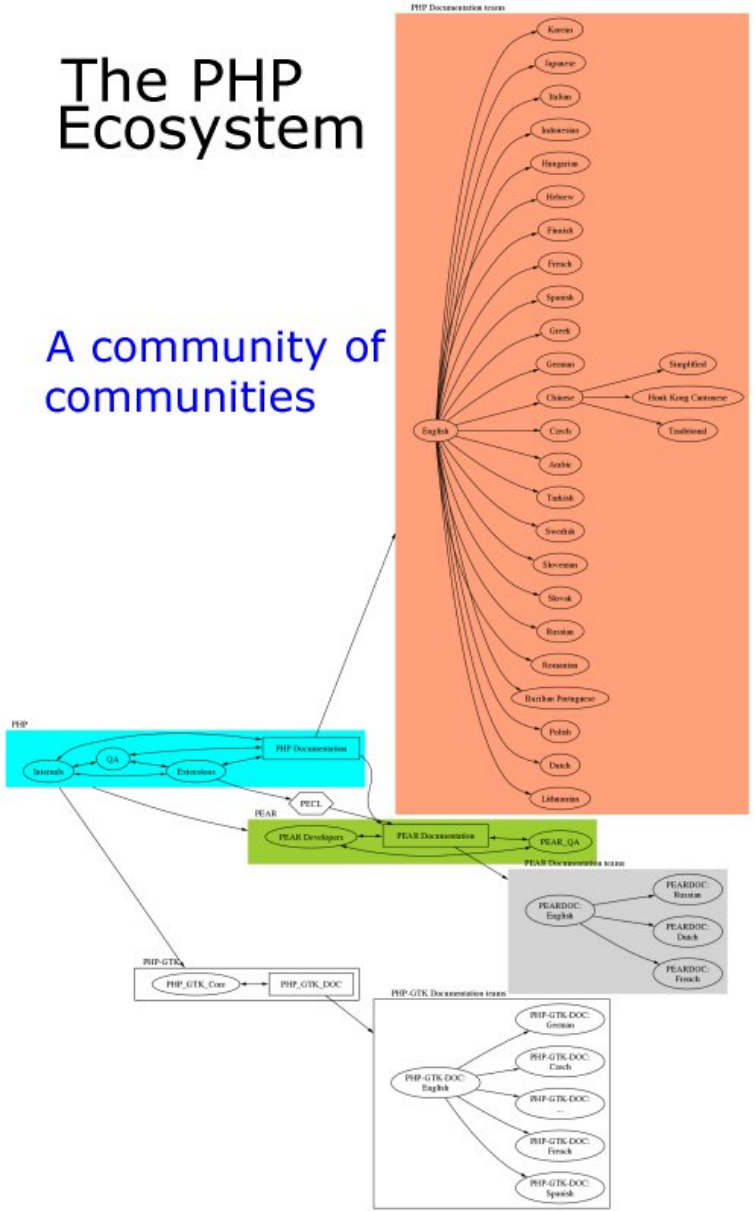
Los desarrolladores de PHP, PEAR, PHP-GTK, y comunidades relacionadas (usuarios, programadores, escritores, etc.), constituyen un ecosistema (artificial). Cada uno se beneficia ("alimenta") de los productos de los otros.

La comunidad de PEAR



The PEAR Communities

... parte de una aún mayor



- Muchos grupos
- Descentralizados
- Superpuestos
- Diferentes culturas
- Múltiples especialidades
- ... un lenguaje (PHP) y una filosofía de trabajo en común (Open Source).

Como se organiza PEAR

- Un grupo grande de desarrolladores con voz y voto (PEAR Dev).
- Grupo de control de calidad (PEAR QA).
- Grupo de documentación (PEAR DOC).
- Grupo central que administra y toma las decisiones acerca del proyecto (PEAR Group).

Estándares de código (1)

- Clases con nombres descriptivos, ej. HTML_Table, XML_RSS, Image_GIS.
- Miembros públicos usan “Camel Caps”, ej. solveEquation(), getStructure().
- Miembros privados se prefijan con subrayado, ej. $_handler, _validate().

Estándares de código (2)

- Constantes usan sólo mayúsculas y se nombran de acuerdo al paquete, ej. MATH_STATS_CUMMULATIVE
- Variables globales deben usar el nombre del paquete con subrayado como inicio, e.j. $GLOBALS['_PEAR_default_error_mode']
- Una clase por archivo, y cada archivo con el nombre de la clase contenida, ej. DB.php

Estándares de código (3)

- Jerarquía plana en la estructura de los directorios, correspondiente a donde será instalado, por ejemplo, HTML_QuickForm_Controller, se instalará en: HTML/QuickForm/Controller
- Líneas de código de 75-85 caracteres.
- Indentado de 4 espacios, no tabulaciones.

Estándares de código (4)

- Las funciones usan el estilo “one true brace”:

```
function getEntry($row, $col)
{
 // cuerpo del método
}
```

- Estructuras de control deben usar llaves siempre:

```
if ($obj->isValid()) {
    // procesarlo
} else {
    // manejar el problema
}
```

Estándares de código (5)

- Sólo se permite usar **<?php ?>** para marcar el bloque de código de PHP.
- Inclusión incondicional de código debe de usar **require_once**, mientras que la inclusión condicional usará **include_once**.
- Comentarios deben usar el estilo de C (/* */) o de C++ (//).

6° Encuentro Linux

Estándares de documentación (1)

- Las clases, miembros públicos, constantes, etc. deberán tener documentación embebida usando usando las convenciones y marcados de **PhpDocumentor**.
- Paquetes considerados estables (**stable**) deberán de proveer documentación para el usuario final, en adición a la generada para su API.

6° Encuentro Linux

Estándares de documentación (2)

- Documentación será generada en forma automática para cada versión, usando PhpDocumentor.
- La documentación que se incluya en el manual se escribe usando Docbook XML (con modificaciones locales).
- Grupos de traductores deberán mantener el manual al día con respecto a la versión en inglés.

6° Encuentro Linux

Ejemplo de código

```php
<?php
/**
 * @package PLUG_Event
 */
/**
 * Class that represents a calendar event for PLUG
 * @author  Jesus M. Castagnetto <jmcastagnetto@php.net>
 * @version 0.0.2
 * @access  public
 */
class PLUG_Event {
    /**
     * Set event's date and time,  and description
     * @param integer $date Event's timestamp
     * @param string   $desc Event's description
     * @return boolean True if OK, False otherwise
     * @access public
     */
     function setEventInfo($date, $desc)
     {
         // code here
     }
}
?>
```

Estado de desarrollo de los paquetes

- Se consideran los siguientes niveles de madurez para un paquete (en orden creciente):
  - **dev** : en desarrollo, inestable.
  - **alpha** : en prueba, API es inestable.
  - **beta** : API casi estable, algunos problemas aún por arreglar.
  - **RC** : API estable, algunos problemas menores pueden existir.
  - **stable** : API estable, todos los problemas solucionados.

Números de versión

- Una versión esta compuesta de un número mayor, uno menor, y uno de nivel de parche, ej. 1.0.2
- El estado (excepto por “stable”) se debe añadir al número de versión, ej. 0.8.3beta2
- El número de versión >= 0.1.0
- Un paquete estable tiene un número de versión >= 1.0.0

Compatibilidad entre versiones

- Si existe una ruptura de la compatibilidad, el nombre del paquete y versión deben reflejarla, ej. PHPUnit2 2.0.0
- Son rupturas de compatibilidad:
  - Cambio de API.
  - Agregado de características.
  - Cambios fundamentales en los requerimientos (ej. necesitar PHP 5 por el uso de Excepciones).

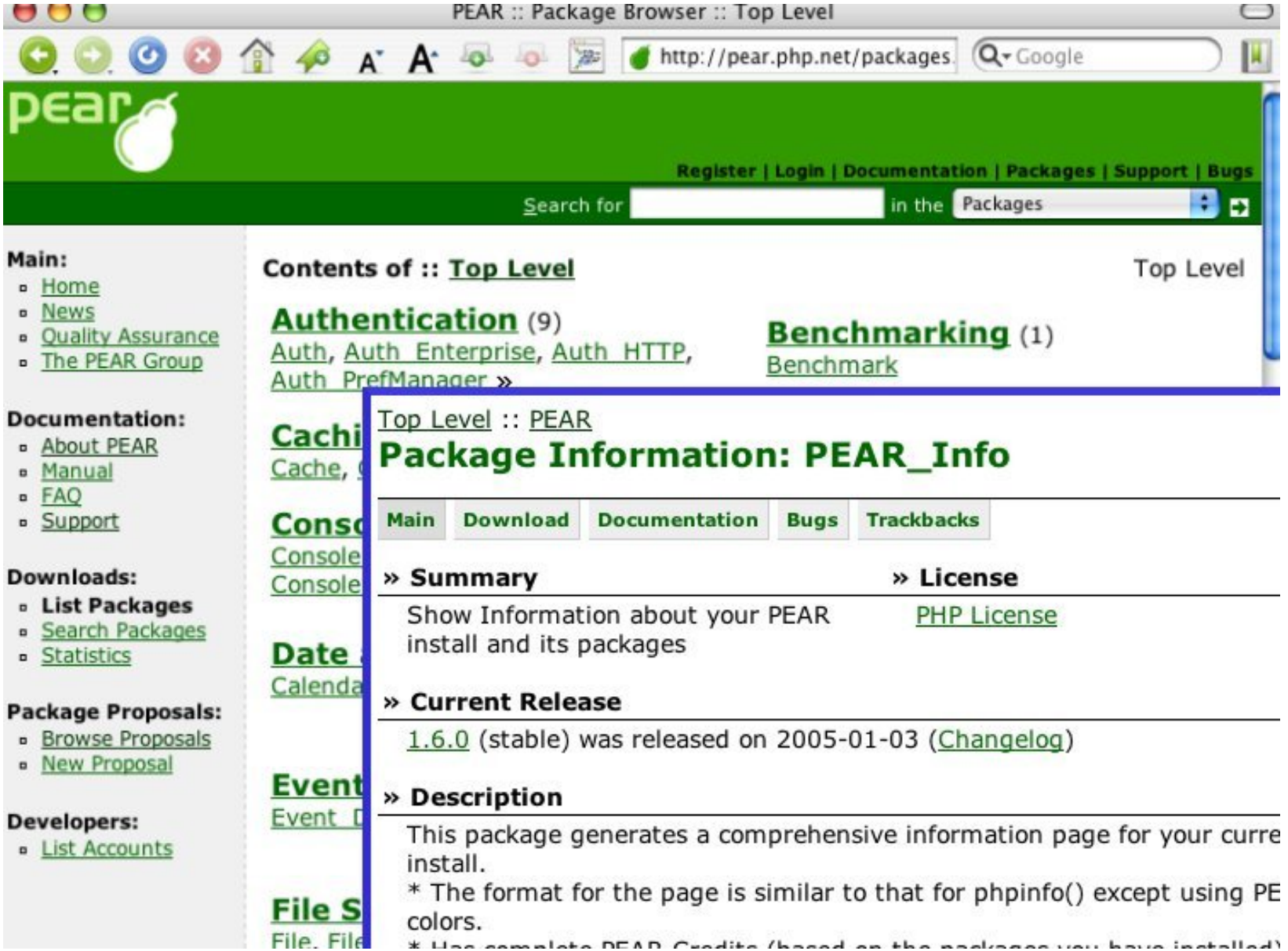
6° Encuentro Linux

Proponiendo un paquete: El sistema PEPr

- PEPr: un sistema basado en la web que permite el proponer, comentar y votar acerca de nuevo código para PEAR.
- Parte de la filosofía de “revisión por los pares” que es esencial en Open Source.
- Pasos bien definidos para nuevo código: de borrador a propuesta a votación.

6° Encuentro Linux

Paquetes en PEAR



El comando **pear**

- Con PEAR, se obtiene el comando 'pear' para manejar paquetes (y extensiones de PECL).
- Hay opciones de comando para:
  - Manejar la configuración local.
  - Instalar, remover, y actualizar paquetes.
  - Conseguir información acerca de paquetes disponibles en repositorios.
  - Construir y validar nuevos paquetes.

6° Encuentro Linux

Obteniendo **pear**

Bajas el código usando:

```
$ wget http://pear.php.net/go-pear
--23:33:49--  http://pear.php.net/go-pear
           => `go-pear'
Resolving pear.php.net... 216.92.131.66
Connecting to pear.php.net[216.92.131.66]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 77,263 [text/plain]

100%[==============================================>] 77,263  8.92K/s
23:33:58 (9.13 KB/s) - `go-pear' saved [77263/77263]
```

... o también:

```
$ curl http://go-pear.org > go-pear
  % Total    % Received % Xferd  Average Speed          Time             Curr.
                                 Dload  Upload Total    Current  Left    Speed
100 77263  100 77263    0     0   5719      0  0:00:13  0:00:13  0:00:00  6987
```

... y finalmente:

```
$ php -f go-pear
Welcome to go-pear! (...)
```

6° Encuentro Linux

Opciones de **pear**

```
$ pear
Commands:
build                   Build an Extension From C Source
bundle                  Unpacks a Pecl Package
channel-add             Add a Channel
channel-alias           Specify an alias to a channel name
channel-delete          Remove a Channel From the List
(... comandos omitidos ...)
uninstall               Un-install Package
update-channels         Update the Channel List
upgrade                 Upgrade Package
upgrade-all             Upgrade All Packages

Usage: pear [options] command [command-options] <parameters>
Type "pear help options" to list all options.
Type "pear help shortcuts" to list all command shortcuts.
Type "pear help <command>" to get the help for the specified command.
```

El comando **pear** tiene mas de 40 opciones

6° Encuentro Linux

Configurando **pear**

Veamos la configuración por defecto:

```
$ pear config-show
CONFIGURATION:
==============
(... lineas omitidas ...)
Preferred Package State        preferred_state stable
Unix file mask                 umask           22
Debug Log Level                verbose         1
HTTP Proxy Server Address      http_proxy      <not set>
PEAR server                    master_server   pear.php.net
(... lineas omitidas ...)
```

Una de las opciones importantes a cambiar, es la que se refiere al estado preferido del paquete. Por defecto es 'stable'.

```
$ pear config-set preferred_state beta
$ pear config-show | grep preferred_state
Preferred Package State        preferred_state beta
```

Listar paquetes con **pear**

```
$ pear list
Installed packages:
===================
Package                      Version State
Archive_Tar                  1.3.1   stable
Auth                         1.2.3   stable
Cache_Lite                   1.5.2   stable
Console_Getopt               1.2     stable
DB                           1.7.6   stable
File_Passwd                  1.1.5   stable
HTML_Common                  1.2.2   stable
HTML_QuickForm               3.2.5   stable
HTML_QuickForm_Controller    1.0.4   stable
HTML_Template_IT             1.1.1   stable
HTML_Template_Sigma          1.1.3   stable
HTTP                         1.3.6   stable
HTTP_Upload                  0.9.1   stable
Mail                         1.1.8   stable
Net_SMTP                     1.2.7   stable
Net_Socket                   1.0.6   stable
PEAR                         1.3.6   stable
XML_Parser                   1.2.6   stable
XML_RPC                      1.4.1   stable
```

- Es fácil ver que tenemos instalado.
- Hay otras interfaces, pero la linea de comando es simple y flexible.

6° Encuentro Linux

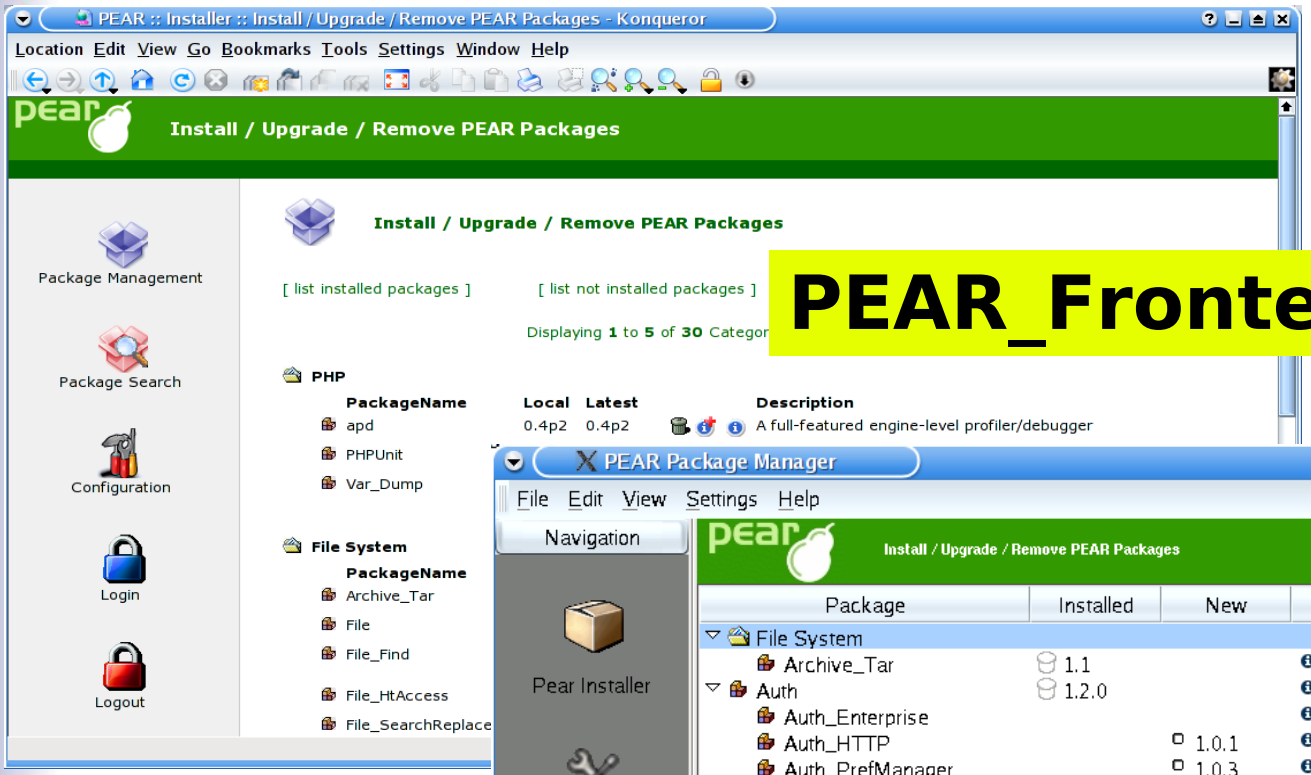
Instalar paquetes con **pear**

Instalemos una GUI para manejar paquetes

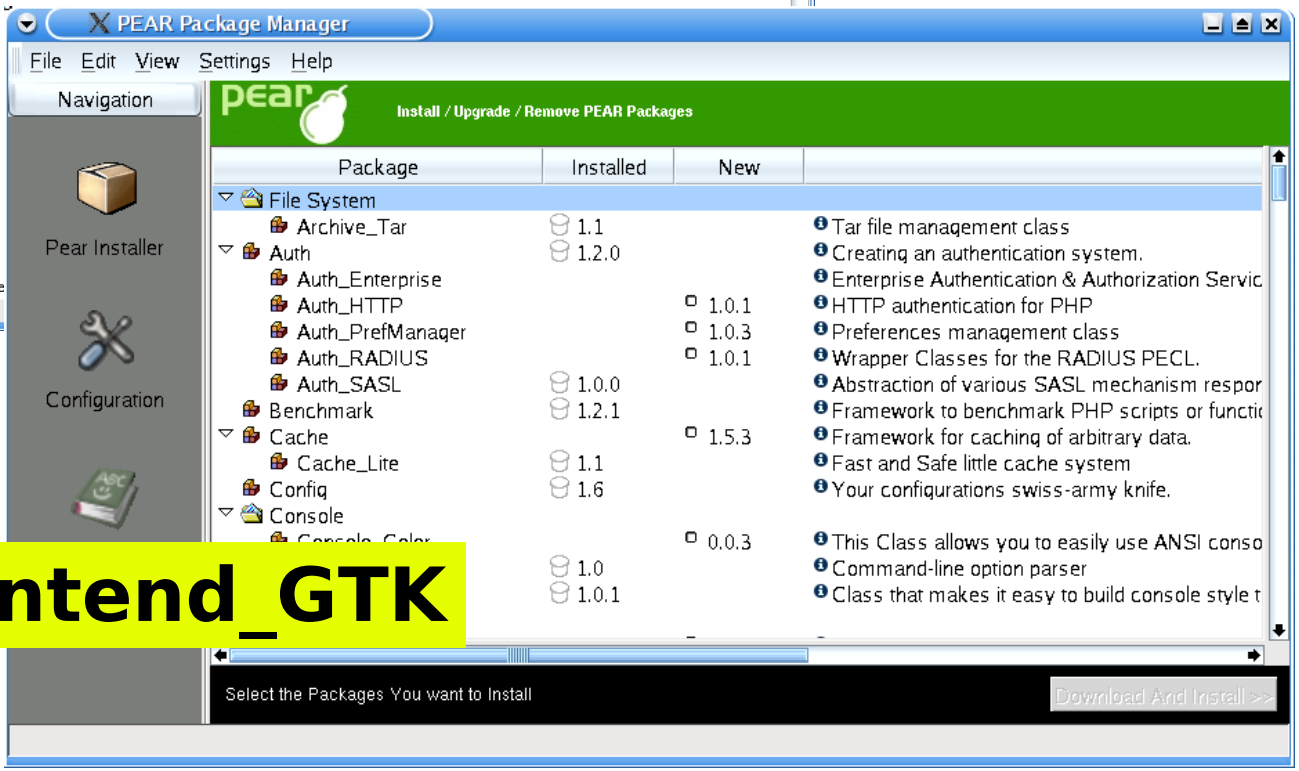
```
$ pear install PEAR_Frontend_Web
downloading PEAR_Frontend_Web-0.4.tgz ...
Starting to download PEAR_Frontend_Web-0.4.tgz (32,386 bytes)
.........done: 32,386 bytes
requires package `Net_UserAgent_Detect'
PEAR_Frontend_Web: Dependencies failed

$ pear install --alldeps PEAR_Frontend_Web
downloading PEAR_Frontend_Web-0.4.tgz ...
Starting to download PEAR_Frontend_Web-0.4.tgz (32,386 bytes)
.........done: 32,386 bytes
downloading Net_UserAgent_Detect-2.1.0.tgz ...
Starting to download Net_UserAgent_Detect-2.1.0.tgz (9,595 bytes)
...done: 9,595 bytes
install ok: Net_UserAgent_Detect 2.1.0
install ok: PEAR_Frontend_Web 0.4
```

El paquete se instaló sin problemas, incluyendo dependencias. Otro paquete que se puede usar es PEAR_Frontend_GTK

Instalar paquetes (2)



PEAR_Frontend_Web



PEAR_Frontend_GTK

Estructura de un paquete

- Un directorio nombrado como el paquete, ej. **`Math_Matrix`**.
- Cada clase en un archivo, reflejando la estructura del paquete, ej. el archivo **`XML_Tree/Tree/Node.php`** contiene la clase **`XML_Tree_Node`**.
- Directorios accesorios conteniendo la documentación, ejemplos, test unitarios y de regresión, etc.

6° Encuentro Linux

Estructura de un paquete (continuación)

- **Paquete_Nombre** : ej. Foo.php
  - **Nombre** : clases sub-componentes del paquete, ej. Modulo.php
  - data : opcional, datos del paquete, ej. lista de TLDs.
  - **docs** : README, RFCs, etc.
    - **examples** : ejemplos de uso
  - misc : opcional, misceláneos.
  - scripts : opcional, ejecutables.
  - **tests** : pruebas de regresión y/o unitarios.

6° Encuentro Linux

Ejemplo: Math_Complex

```
Math_Complex
   |-- Complex.php
   |-- ComplexOp.php
   |-- docs
   |    |-- Math_Complex.dia
   |    |-- Math_Complex.png
   |    |-- README
   |    `-- examples
   |         `-- using_complexop.php
   |-- package.xml
   `-- tests
        `-- phpunit
             |-- unitTest_Math_Complex.php
             `-- unitTest_Math_ComplexOp.php
```

6° Encuentro Linux

Escribiendo un paquete: MondoRegex

Una sola clase que simplemente hace multiples regexes

```
<?php
    class MondoRegex {
        var $_reList = array;
        var $_matchList = array();

        function MondoRegex($reList = "") {}
        function setRegexList($reList) {}
        function match($inputList) {}
        function matchAll($inputList) {}
        function getMatches() {}
        function getMatchesForKey($reName) {}
        function getMatchKeys() {}
    }
?>
```

6° Encuentro Linux

El archivo **package.xml**

- Se usa para definir los roles de cada componente de un paquete.
- Define dependencias requeridas y opcionales.
- Da un sumario de un paquete, la licencia, y los responsables de mantener y desarrollarlo.
- Contiene un listado de cambios de la versión del paquete.

6° Encuentro Linux

Elementos de **package.xml**

- **package** – define el paquete, opcionalmente con una versión
  - **name** – El nombre del paquete
  - **summary** – sumario de una línea
  - **description** – descripción completa del paquete
  - **license** – la licencia a usarse (ej. PHP, LGPL, etc.)
  - **maintainers** – lista de mantenedores
    - **maintainer** – contenedor para un mantenedor
    - **user** – nombre de usuario
    - **role** – su rol: lead, developer, etc.
    - **name** – nombre completo
    - **email** – dirección de correo electrónico
  - **release** – contenedor para una versión
    - **version** – número de versión, ej. 1.0.2
    - **state** – estado del paquete (dev, alpha, beta, RC, stable)
    - **date** – fecha de emisión
    - **notes** – notas acerca de la versión
    - **filelist** – lista de archivos
      - **file** – archivo, con atributos indicando su rol, etc.
      - **dir** – directorio (puede contener mas elementos **file**)
    - **deps** – lista de dependencias
      - **dep** – dependencia, indicando tipo y si es o no opcional
  - **changelog** – lista de cambios por cada versión

package.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE package SYSTEM "../package.dtd">
<package>
 <name>MondoRegex</name>
 <summary>Class to apply a list of regular expressions</summary>
 <description>
  This class applies an array of named regular expressions
  to an array of input strings.
 </description>
 <maintainers>
  <maintainer>
   <user>jmcastagnetto</user>
   <name>Jesus M. Castagnetto</name>
   <email>jmcastagnetto@php.net</email>
   <role>lead</role>
  </maintainer>
 </maintainers>
 <release>
  <license>PHP</license>
  <version>0.1.0</version>
  <date>2003-06-05</date>
  <notes>Initial release</notes>
  <state>beta</state>
  <filelist>
   <dir name="/" baseinstalldir="JMC">
    <file role="php">MondoRegex.php</file>
   </dir>
  </filelist>
 </release>
 <changelog>
 </changelog>
</package>
```

Creando **package2.xml**

- Podemos crearlo a mano
- O quizás configurar y usar el paquete PEAR_PackageFileManager
- O, usar el comando **pear** para ayudarnos, y editar el resultado:

```
$ pear convert package.xml package2.xml
Wrote new version 2.0 package.xml to
"./package2.xml"
$ vim package2.xml (... etc.)
```

package2.xml

```
<?xml version="1.0"?>
<package packagerversion="1.4.0a12" version="2.0"
xmlns="http://pear.php.net/dtd/package-2.0"
xmlns:tasks="http://pear.php.net/dtd/tasks-1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://pear.php.net/dtd/tasks-1.0
http://pear.php.net/dtd/tasks-1.0.xsd
http://pear.php.net/dtd/package-2.0
http://pear.php.net/dtd/package-2.0.xsd">
 <name>MondoRegex</name>
 <channel>pear.castagnetto.org</channel>
 <summary>Class to apply a list of regular expressions</summary>
 <description>This class applies an array of named
regular expressions to an array of input strings.
 </description>
 <lead>
  <name>Jesus M. Castagnetto</name>
  <user>jmcastagnetto</user>
  <email>jmcastagnetto@php.net</email>
  <active>yes</active>
 </lead>
 <date>2005-10-22</date>
 <time>01:02:20</time>
(... etc ...)
</package>
```

6° Encuentro Linux

Preparando el paquete

```
$ pear package-validate
Validation: 0 error(s), 0 warning(s)

$ pear package package.xml package2.xml
Attempting to process the second package file
Analyzing MondoRegex.php
Package MondoRegex-0.1.0.tgz done

$ pear install MondoRegex-0.1.0.tgz
install ok: MondoRegex 0.1.0

$ php -f test.php
TESTING MondoRegex
Using match
...

$ cp MondoRegex-0.1.0.tgz /path/to/server/packages/
```

Creando un canal de distribución de paquetes

- Requerimientos:
  - PHP 5 (version >= 5.0.3)
  - PEAR 1.4.x
  - MySQL
  - PEAR_Server
- Los módulos a distribuir deben de tener también un archivo de descripción de paquetes versión 2 (package2.xml).

Creando un canal de distribución (cont.)

- Registramos el servidor de Greg Beaver:

```
$ pear channel-discover pear.chiaraquartet.net
```

- Instalamos el PEAR_Server

```
$ pear install –alldeps chiara/PEAR_Server
```

- El instalador le dirá que se necesita correr un script de post-instalación:

```
$ pear run-scripts PEAR_Server
```

  y se contestan las preguntas acerca del servidor, credenciales, base de datos, etc.

6° Encuentro Linux

Creando un canal de distribución (cont.)

- Renombramos el archivo de administración (**frontend.php**), que usaremos para manejar nuestro canal

```
$ cd /dir/del/server/htdocs
$ mv frontend.php _myadmin-interface.php
```

- Creamos un index.html (o index.php) con información acerca del canal.
- Auto-descubrimos nuestro canal

```
$ pear channel-discover pear.example.com
```

- Nos conectamos al servidor, y via la interfaz de web, agregando nuestro nuevo paquete, mantenedor(es), etc.

Usando el nuevo canal

- Todo está listo para que la gente registre nuestro canal:

```
$ pear channel-discover pear.example.com
```

- Y empieze a instalar los paquetes:

```
$ pear micanal/MondoRegex
```

o

```
$ pear pear.example.com/MondoRegex
```

- Lo que resta hacer es documentación de nuestros paquetes, soporte, etc.

6° Encuentro Linux

Para más información

- El sitio de PEAR:
  http://pear.php.net/
- La lista/archivo de PEAR-dev.
- El blog de Greg Beaver:
  http://greg.chiaraquartet.net/
- El blog de Tobias Schlitt:
  http://www.schlitt.info/applications/blog/
- El blog de Davey:
  http://pixelated-dreams.com/