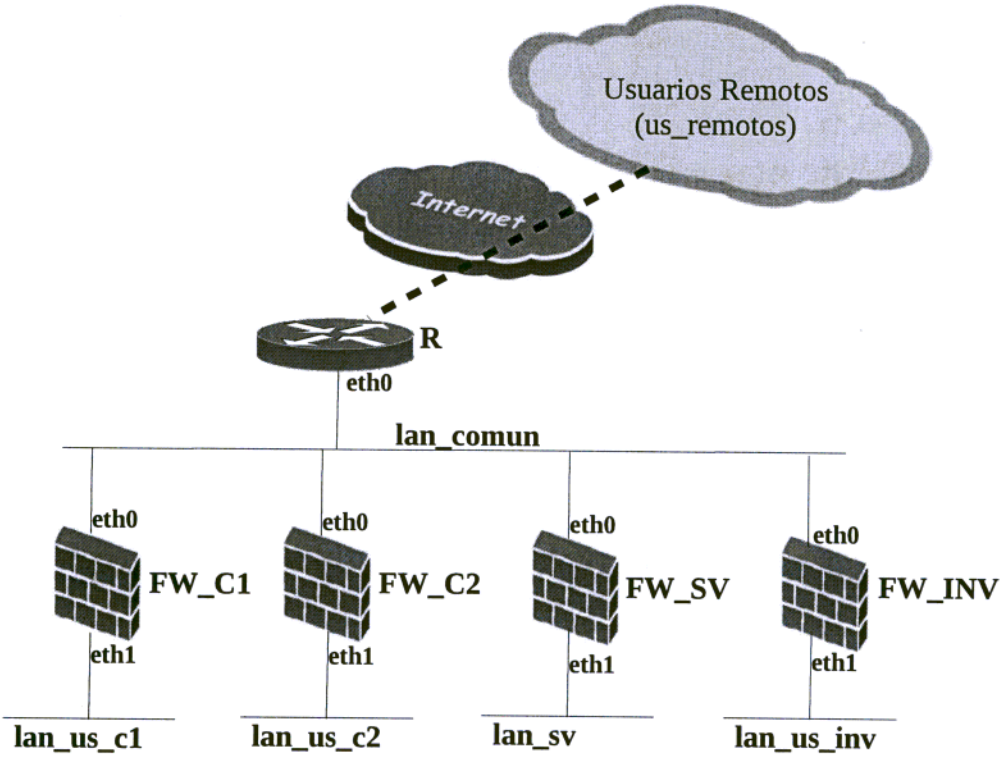
Pruebas selectivas para el acceso a puestos vacantes de personal funcionario del Cuerpo Técnico de la Administración de la Comunidad Autónoma de Extremadura. Publicadas en Orden 27/12/2013.

Tribunal nº: 6. Turnos Libre y Discapacidad. Especialidad: Informática. Segunda fase del ejercicio

SUPUESTO 1 (2,50 Puntos)

La Junta de Extremadura ha construido un nuevo edificio que va a servir de sede para usuarios de dos Consejerías diferentes. A nivel de red, se ha establecido la topología mostrada en el siguiente esquema:



Descripción de la topología de red:

- La sede dispondrá de un router (R) para el enrutamiento del tráfico cursado con Internet y entre los distintos segmentos de red entre si.
- A través del router R se permitirá la conexión de usuarios remotos mediante el establecimiento de túneles IPSec.
- Se dispone de una electrónica de red que permite el establecimiento de diferentes segmentos de red mediante la implantación de diferentes redes virtuales (VLAN)
- También se dispone de dispositivos de seguridad que permiten la implementación de distintos firewalls y que permiten enrutar y filtrar el tráfico entre las distintas VLAN.
- Los diferentes firewalls mostrados realizan el enrutado básico entre el tráfico de la lan que protegen y el segmento de "lan_comun". La puerta de enlace de todos los firewall será la IP configurada en la interfaz "eth0" del router "R". De esta forma, será el router "R" el encargado de enrutar el tráfico entre los distintos segmentos de red mediante la configuración de las rutas estáticas que sean necesarias.

- Conforme a lo anterior se han establecido distintas VLAN. La tabla 1 muestra la descripción de cada una de las VLAN y la previsión del número de dispositivos IP (ordenadores, impresoras, dispositivos móviles, etc...) que se conectarán a cada una de ellas.
- En cuanto al número de usuarios remotos (us_remotos) se debe establecer un dimensionamiento que permita la conexión de al menos 20 usuarios (consumiendo una dirección IP cada uno de ellos).
- El administrador de la red ha asignado para el conjunto de esta sede el espacio de direccionamiento siguiente: 172.24.0.0/22 (172.24.0.0/255.255.252.0)

Nombre VLAN	Descripción	Previsión del número de dispositivos IP a considerar como mínimo.
lan_comun	Segmento de red a la que se conecta el router R y cada uno de los firewall que separan las distintas VLAN.	20
lan_us_c1	Segmento de red (vlan) a la que se conectarán los usuarios pertenecientes a la Consejería 1	320
lan_us_c2	Segmento de red (vlan) a la que se conectarán los usuarios pertenecientes a la Consejería 2	210
lan_sv	Segmento de red (vlan) donde se conectarán los servidores de la sede.	90
lan_us_inv	Segmento de red (vlan) donde se conectarán los usuarios invitados (trabajadores pertenecientes a empresas colaboradoras que desarrollan su actividad de forma temporal en la sede)	50

Tabla 1: Descripción y dimensionamiento de las distintas VLAN.

1.1. (0,90 Puntos)

Usando el rango de red 172.24.0.0/22, cumplimentar una tabla, con el formato mostrado a continuación, asignando una subred para cada uno de los segmentos de red indicados. Cada subred asignada deberá tener el número de direcciones IP necesarias para permitir, al menos, la conexión de los dispositivos IP mostrados en la tabla 1 así como los 20 usuarios remotos ya indicados.

NOMBRE VLAN	Nº de dir. IP que debe contener la subred asignada como mínimo	Red	Mascara	Nº total de direcciones IP contenidas en la subred asignada
.....	.....	.....	.....	.....
.....	.....	.....	.....	.....
.....	.....	.....	.....	.....

1.2. (0,50 Puntos)

De forma coherente con el enunciado del supuesto y con la asignación realizada en el apartado anterior, cumplimentar una tabla con el formato mostrado a continuación, indicando la configuración a aplicar en las distintas interfaces de red "eth0" de cada uno de los firewalls, así como el resto de parametros IP indicados (mascara de red, dirección de red, broadcast y puerta de enlace).
Nota: no se debe rellenar el campo "puerta de enlace" correspondiente al router R

Equipo	IP eth0	Mascara de red	Dirección de red	Broadcast	Puerta de enlace
.....	.....	.....	.....	.....	----------
.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....

1.3. (0,80 Puntos)

De nuevo, de forma coherente con lo indicado en los apartados anteriores, señalar las rutas estáticas que sería necesario configurar en el router R para que el tráfico a las siguientes redes (VLAN) sea enrutado correctamente: "lan_us_c1", "lan_us_c2", "lan_sv" y "lan_us_inv".
Para señalar las rutas utilizar el comando "route" de Linux:

route add [-net|-host] target [netmask Nm] [gw Gw]

1.4. (0,30 Puntos)

Suponiendo que el cortafuegos "FW_SV" se configura mediante sentencias "*iptables*", indica la sentencia o sentencias "*iptables*" que se establecerían en "FW_SV" para activar como política por defecto que se descarte todo el tráfico procedente de cualquier origen y con destino cualquier dispositivo perteneciente a "lan_sv". *Ver ayuda iptables anexo 1.*

SUPUESTO 2 (1,50 Puntos)

Como técnico del área de sistemas se le ha encargado la realización de las siguientes tareas:

2.1. (0,75 Puntos)

Codificar un shell script para que compruebe si un cierto numero de puerto TCP que se le ha pasado como parámetro esta en estado de escucha (LISTENING). El shell script sacará por pantalla información indicando si el puerto pasado por parámetro esta en dicho estado o no. El shell script tiene que aceptar como parámetro un número de puerto entre 1 y 1024, en caso contrario sacará un mensaje de error y finalizará.

2.2. (0,75 Puntos)

Codificar un shell script en el que detecte la ejecución (**estado R**) de un proceso llamado **backupd** y que envíe mediante un email a **alertas@cpd.pri** información que indique si dicho proceso se esta ejecutando o no.

SUPUESTO 3 (3,50 Puntos)

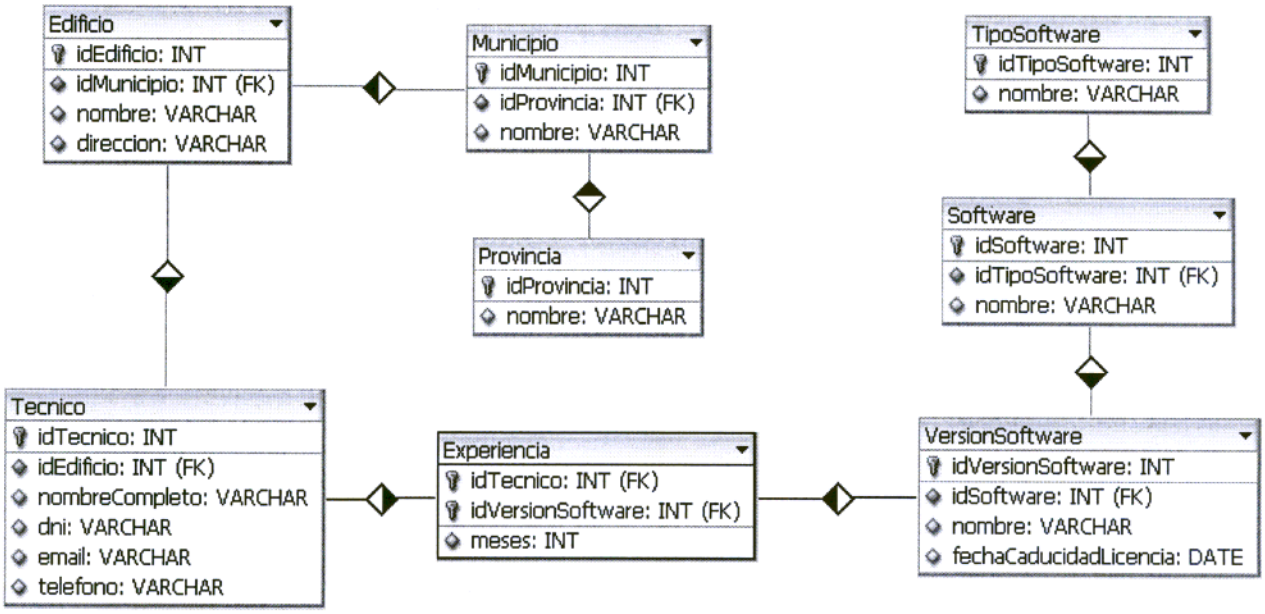


Figura 1. Diagrama de base de datos

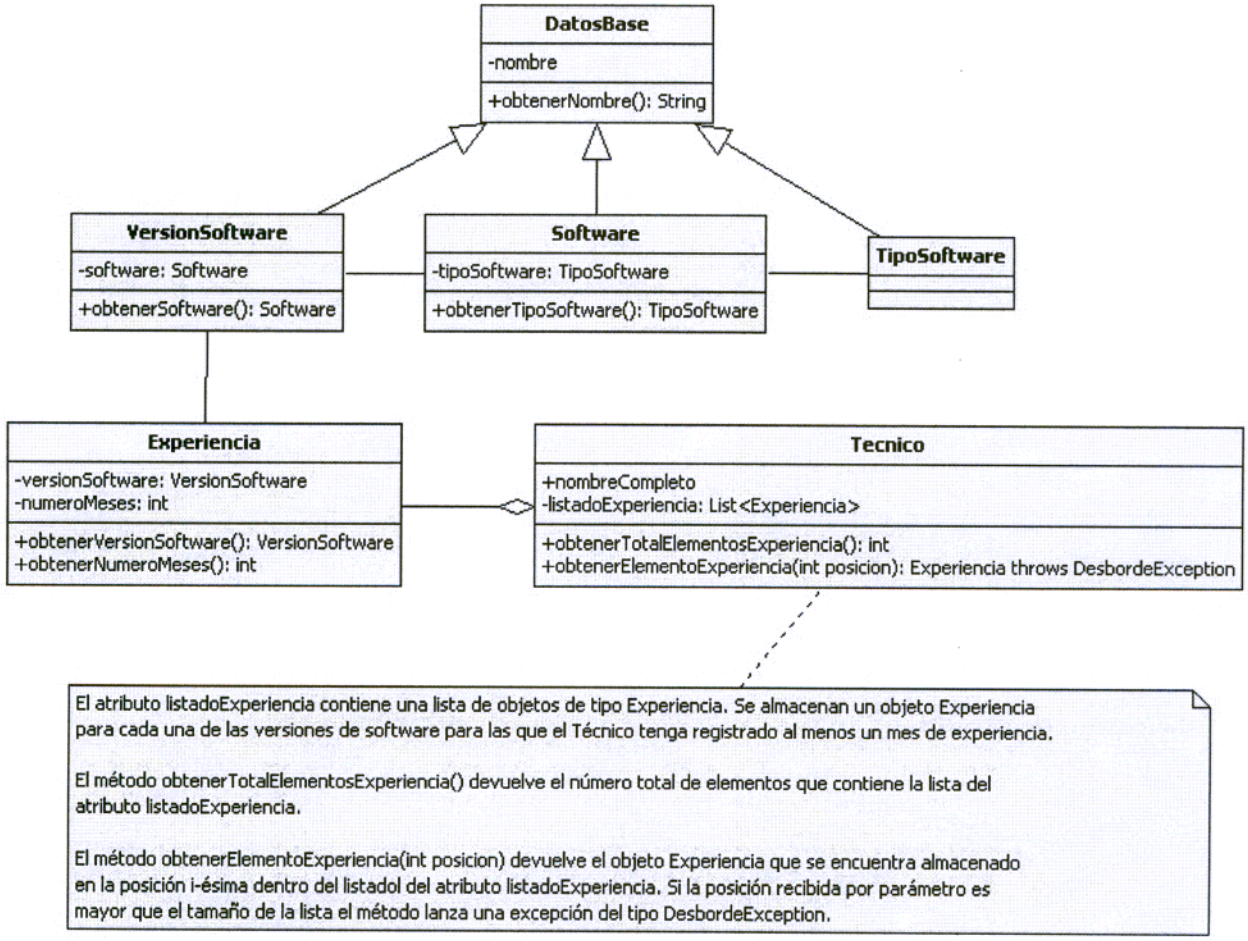


Figura 2. Diagrama de clases

El Servicio de Desarrollo de Software de la Junta de Extremadura, con el objetivo de optimizar los recursos humanos de los que dispone, está trabajando en la puesta en marcha de un sistema de información que, entre otras funcionalidades, facilite la selección y formación de los equipos de trabajo asignados a cada uno de los proyectos y/o sistemas que son competencia de dicho Servicio.

Uno de los módulos del sistema se encarga exclusivamente de registrar y almacenar información relativa a la experiencia que los programadores van acumulando durante su carrera profesional respecto al manejo de las distintas herramientas existentes en el mercado. En primer lugar, se ha definido una estructura jerárquica que posibilita la clasificación de la experiencia en función de la naturaleza de las herramientas software usadas. Para ello, se han definido *Tipos de Software* (Servidores, BasesDatos, etc), se han catalogado las distintas herramientas *Software* existentes (Jboss, Tomcat, IIS, SQL Server, Oracle, etc) así como las *Versiones* concretas de cada una de las herramientas software dadas de alta en el sistema.

En la figura 1, de la página anterior, se expone el diagrama de la base de datos del módulo definido en el párrafo anterior y en la figura 2, un diagrama de clases java proporcionado por el analista.

Basándose en todo lo anteriormente expuesto, se pide:

3.1. (1 Punto)

Dado el esquema de base de datos anterior, escribir una sentencia en SQL para eliminar todos las versiones de software que sean del tipo 'Servidores' (siendo 'Servidores' un nombre de Tipo Software) y cuya fecha de caducidad de la licencia sea durante el año 2016.

3.2. (1 Punto)

Dado el esquema de base de datos anterior, escribir una sentencia en SQL para obtener un listado de técnicos que contenga la siguiente información: nombre, teléfono y total de meses de experiencia acumulada en cualquiera de las versiones de software registradas en la bbdd para el software *'Tomcat'* (siendo '*Tomcat*' un nombre de Software). Además, en el listado solo se mostrarán aquellos técnicos que tengan al menos un total de dos años de experiencia en dichas versiones de software y aparecerá ordenado de forma que en primer lugar aparezca el técnico con más experiencia.

3.3. (0,50 Puntos)

Dado el diagrama de clases Java anterior, indicar el código Java necesario para definir una interfaz, denominada *GestorExperiencia*, que herede de una interfaz denominada *GestorBase* y que contenga un único método denominado *listarExperienciaTecnico*. Dicho método debe definirse de forma que reciba un objeto de la clase *Tecnico* como único parámetro de entrada y no devuelva ningún objeto.

3.4. (1 Punto)

Dado el diagrama de clases Java anterior, indicar el código Java necesario para implementar una clase, denominada *GestorExperienciaImpl*, que implemente la interfaz *GestorExperiencia* del apartado anterior. Así, la funcionalidad del método *listarExperienciaTecnico* consistirá en mostrar por consola toda la información relativa a la experiencia del técnico que se recibe por parámetro. Así, se deberá mostrar por consola algo similar a:

Técnico: Juan González García
- *Software: JBoss*
- *Versión: Jboss 6 EAP*
- *Experiencia: 6 meses*

- *Software: SQL Server*
- *Versión: SQL Server 2008 R2*
- *Experiencia: 18 meses*

SUPUESTO 4 (1 Punto)

Desde el área de desarrollo se le ha encargado la codificación de una página web en PHP para que lea el fichero que contiene las unidades de proceso (**ups.xml)** y muestre el contenido de los elementos **up**. El fichero **ups.xml** está bien formado y puede contener más de un elemento **up.**

El dtd que valida el fichero **ups.xml** es:

```
<!DOCTYPE ups
[
<!ELEMENT ups (up+)>
<!ELEMENT up (nombre,fqdn,ip)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT fqdn (#PCDATA)>
<!ELEMENT ip (#PCDATA)>
]>
```

Un ejemplo de un fichero válido **ups.xml** sería:

```
<?xml version="1.0" encoding="UTF-8"?>


<!DOCTYPE ups
[
<!ELEMENT ups (up+)>
<!ELEMENT up (nombre,fqdn,ip)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT fqdn (#PCDATA)>
<!ELEMENT ip (#PCDATA)>
]>

<ups>
  <up>
    <nombre>Servidor1</nombre>
    <fqdn>Servidor1.cpd.pri</fqdn>
    <ip>172.30.20.10</ip>
  </up>
  <up>
    <nombre>Servidor2</nombre>
    <fqdn>Servidor2.cpd.pri</fqdn>
    <ip>172.30.20.11</ip>
  </up>
</ups>
```

Como ayuda, en el anexo 2 se detalla la sintáxis de la función ***simplexml_load_file*** *que se podrá usar en la codificación.*

SUPUESTO 5 (1,50 Puntos)

La notación polaca inversa, es un método algebraico alternativo de introducción de datos. En ella primero están los operandos y después viene el operador que va a realizar los cálculos sobre ellos. La expresión algebraica **(3+7)*2** se traduce a la notación polaca inversa como **3 7 + 2 *** y se evalúa de izquierda a derecha. Realiza un algoritmo en pseudocódigo que resuelva expresiones algebraicas dadas en notación polaca inversa.

PUNTACIÓN ASIGNADA A CADA SUPUESTO	
Supuesto 1	**2,50 Puntos**
Apartado 1	0,90 Puntos
Apartado 2	0,50 Puntos
Apartado 3	0,80 Puntos
Apartado 4	0,30 Puntos
Supuesto 2	**1,50 Puntos**
Apartado 1	0,75 Puntos
Apartado 2	0,75 Puntos
Supuesto 3	**3,50 Puntos**
Apartado 1	1,00 Punto
Apartado 2	1,00 Punto
Apartado 3	0,50 Puntos
Apartado 4	1,00 Punto
Supuesto 4	**1 Punto**
Supuesto 5	**1,50 Puntos**

NOTA: ***El opositor podrá realizar aquellas suposiciones adicionales a los enunciados que considere oportunas, indicándolas convenientemente y justificándolas.***

Anexo 1: (Sintáxis iptables)

Extraido de Man IPTABLES (http://ipset.netfilter.org/iptables.man.html)

SYNOPSIS

iptables [-t table] {-A|-C|-D} chain rule-specification

iptables [-t table] -I chain [rulenum] rule-specification

iptables [-t table] -R chain rulenum rule-specification

iptables [-t table] -X [chain]

iptables [-t table] -P chain target

TARGETS

A firewall rule specifies criteria for a packet and a target. If the packet does not match, the next rule in the chain is the examined; if it does match, then the next rule is specified by the value of the target, which can be the name of a user-defined chain or one of the special values ACCEPT, DROP, QUEUE or RETURN.

ACCEPT means to let the packet through. DROP means to drop the packet on the floor. QUEUE means to pass the packet to userspace. (How the packet can be received by a userspace process differs by the particular queue handler. 2.4.x and 2.6.x kernels up to 2.6.13 include the ip_queue queue handler. Kernels 2.6.14 and later additionally include the nfnetlink_queue queue handler. Packets with a target of QUEUE will be sent to queue number '0' in this case. Please also see the NFQUEUE target as described later in this man page.) RETURN means stop traversing this chain and resume at the next rule in the previous (calling) chain. If the end of a built-in chain is reached or a rule in a built-in chain with target RETURN is matched, the target specified by the chain policy determines the fate of the packet.

TABLES

There are currently five independent tables (which tables are present at any time depends on the kernel configuration options and which modules are present).

-t, --table table

This option specifies the packet matching table which the command should operate on. If the kernel is configured with automatic module loading, an attempt will be made to load the appropriate module for that table if it is not already there.

The tables are as follows:

filter: This is the default table (if no -t option is passed). It contains the built-in chains INPUT (for packets destined to local sockets), FORWARD (for packets being routed through the box), and OUTPUT (for locally-generated packets).

nat: This table is consulted when a packet that creates a new connection is encountered. It consists of three built-ins: PREROUTING (for altering packets as soon as they come in), OUTPUT (for altering locally-generated packets before routing), and POSTROUTING (for altering packets as they are about to go out).

OPTIONS

-A, --append chain rule-specification

Append one or more rules to the end of the selected chain. When the source and/or destination names resolve to more than one address, a rule will be added for each possible address combination.

-C, --check chain rule-specification

Check whether a rule matching the specification does exist in the selected chain. This command uses the same logic as -D to find a matching entry, but does not alter the existing iptables configuration and uses its exit code to indicate success or failure.

-D, --delete chain rule-specification

-D, --delete chain rulenum

Delete one or more rules from the selected chain. There are two versions of this command: the rule can be specified as a number in the chain (starting at 1 for the first rule) or a rule to match.

-I, --insert chain [rulenum] rule-specification

Insert one or more rules in the selected chain as the given rule number. So, if the rule number is 1, the rule or rules are inserted at the head of the chain. This is also the default if no rule number is specified.

-P, --policy chain target

Set the policy for the chain to the given target. See the section TARGETS for the legal targets. Only built-in (non-user-defined) chains can have policies, and neither built-in nor user-defined chains can be policy targets.

Anexo 2: (Sintáxis de la función simplexml_load_file)

simplexml_load_file

(PHP 5)

simplexml_load_file — Interpreta un fichero XML en un objeto

Descripción

SimpleXMLElement **simplexml_load_file** (string `$filename` [, string `$class_name` = "SimpleXMLElement" [, int `$options` = 0 [, string `$ns` = "" [, bool `$is_prefix` = false]]]])

Convierte el documento correcto XML del fichero dado en un objeto.

Parámetros

`filename`

Ruta al fichero XML

Nota:

Libxml 2 no escapa la URI, así que si es necesario pasar, por ejemplo *b&c*, como parámetro *a* de la URI, hay que llamar a *simplexml_load_file(rawurlencode('http://example.com/?a=' . urlencode('b&c')))*. Desde PHP 5.1.0 no es necesario hacer esto porque PHP lo hará automáticamente.

`class_name`

Este parámetro opcional puede ser usado cuando se necesita que **simplexml_load_file()** retorne un objeto de la clase especificada. Esa clase debe extender de la clase SimpleXMLElement.

`options`

Desde PHP 5.1.0 y Libxml 2.6.0 puede usarse `options` para especificar parámetros Libxml adicionales.

`ns`

El prefijo del espacio de nombres o un URI.

`is_prefix`

TRUE si ns es un prefijo, FALSE si es un URI; por defecto es FALSE.

Valores devueltos

Retorna un object de tipo SimpleXMLElement cuyas propiedades contienen los datos del documento XML, o FALSE en caso de error.

Errores/Excepciones

Produce un mensaje de error E_WARNING para cada error encontrado en los datos XML.

Sugerencia

Usar libxml_use_internal_errors() para suprimir todos los errores XML y libxml_get_errors() para iterar sobre cada uno de ellos posteriormente.

Ejemplos

Ejemplo #1 Interpreta un documento XML

```
<?php
//El fichero test.xml contiene un documento XML con un elemento raíz
y,al
//menos,un elemento/[raiz]/titulo.

If (file_exists('test.xml')) {
 $xml = simplexml_load_file('test.xml');

 print_r($xml);
} else {
 exit('Error abriendo test.xml.');
}
?>
```

Este script mostrará, si no hay errores:

```
SimpleXMLElement Object
(
  [titulo] => Título ejemplo
  ...
)
```

En este punto, se puede seguir recorriendo usando *$xml->titulo* y cualquier otro elemento.