Capítulo 5 - Modificación - Actualización de cliente

Modificación - Actualización de cliente

Objetivo: Una aplicación web para enumerar y actualizar la información del cliente.
Dificultad: ***

Preparación

Duplicar archivo `lista_de_clientes.php` desde la Sección 3.3 a un archivo llamado `U-CRUD.php` en la raíz de su servidor web.

La actualización de los recursos a través del servicio web es complejo, por lo que explicaremos en primer lugar su funcionamiento.

Podemos ver que el diagrama está dividido en 2 etapas:

1. Conseguir el recurso a un id definido (1 in the diagram) y crear el formulario.
2. Actualización de recursos.

La flecha apunta a "get", lo que corresponde a un recurso a obtener.
Este paso es importante porque tenemos que obtener el archivo XML para que coincida con los datos enviados por el formulario antes de que podamos llamar a "edit" para actualizar el recurso.

Tenga en cuenta que pudieramos haber modificado de otro modo mediante el envío de un documento XML utilizando Javascript y por lo tanto, no haber utilizado "get" en este proceso.

Paso 1: Obtención de datos y creación del formulario

Recuperar el archivo XML y mostrar el formulario:

```
// Define el recurso
$opt = array( 'resource' => 'customers' );
// Definie el ID del recurso a modificar
$opt[ 'id' ] = $_GET[ 'id' ];
// Llama al servicio web, recuperar el archivo XML
$xml = $webService->get( $opt );
// Recupera elementos de recurso en una variable (table)
$resources = $xml->children()->children();
// formulario del cliente
```

En este caso, la llamada es similar a la obtención de datos. Esta es la llamada que nos permitirá crear el formulario.

Vamos a generar el formulario de actualización automática.

Por el momento, usar etiquetas HTML "input" como su "name" el nombre del atributo, y como su "value" el valor del atributo.

Con el fin de no perder el id del segundo paso de acuerdo con el diagrama, el formulario se muestra como: `?id = "ID de cliente"`
Por lo tanto obtendremos esto: `$_GET[ 'id' ];`

Podríamos haber hecho esto de manera diferente, por ejemplo, mediante la aprobación de esta identificación en POST, pero verá que este método va a simplificar el proceso que sigue.

Paso 2: Actualización del recurso

Inicialmente, como se puede ver en la flecha "Note" en el diagrama, vamos a recuperar el archivo XML. Para ello, se llevará a cabo la misma llamada que se hizo al crear el formulario.

Si ha especificado, como se indicó anteriormente, el destino del formulario con un id, su llamada debería haberse realizado y el formulario se vuelve a mostrar.

Ayuda para la creación de un formulario:

```
foreach ( $resources as $key => $resource ){
  echo '<tr><th>' . $key . '</th><td>';
  echo '<input type="text" name="' . $key . '" value="' . $resource . '"/>';
  echo '</td></tr>';
}
```

Una vez que el archivo XML se recupera tenemos que modificar los nuevos datos con los datos recibidos por POST.

Ruta de las claves en el archivo XML y valores de actualización:

```
foreach ( $resources as $nodeKey => $node ) {
  $resources->$nodeKey = $_POST[ $nodeKey ];
}
```

Ahora tenemos un archivo XML actualizado. Ahora sólo tenemos que enviarlo.

Ejemplo de una actualización:

```
$opt = array( 'resource' => 'customers' ); // Definición del recurso
$opt[ 'xml' ] = $xml->asXML(); //Definición del archivo XMl
$opt[ 'id' ] = $_GET[ 'id' ]; // Definición del ID a modificar
// Calling asXML () returns a string corresponding to the file
$xml = $webService->edit( $opt ); // Call
```

Ahora, en su secuencia de comando `U-CRUD.php`, trate de modificar un cliente con un ID definido en el códigoy luego hacerlo para todos los clientes.

Revise usando `R-CRUD.php` ha cambiado y luego procesar el ID de cliente.

Si tiene problemas, observe el código `2-update.php`.