GUÍA DE PROGRAMACIÓN

Sistema de Control de Producción

SICPROD V1.0

Autor: Joffre Danilo Vásquez Núñez

Año: 2012

GUÍA DE PROGRAMACIÓN

1. Estándares de programación

Es muy importante al momento de emprender un proyecto la estandarización de normas y políticas que permitan la mejor comprensión de los documentos, código de programación, implementación de la base de datos y demás recursos inmersos, entre las personas relacionadas en el desarrollo.

Este documento pretende dar los lineamientos necesarios que permitan un mejor entendimiento de la codificación implementada en el diseño y desarrollo del Sistema de Control de Producción.

1.1 Propósito

El presente documento tiene como finalidad dar a conocer a los interesados los estándares de programación que regirán el desarrollo y mantenimiento de la aplicación que se desea implementar, el mismo que servirá de base para el desarrollo de aplicaciones futuras.

1.2 Descripción

El presente documento muestra al interesado las reglas y normativas que permita estandarizar el desarrollo del Proyecto “Sistema de Control de Producción”, utilizando el framework Javascript EXT JS, PHP como lenguaje del lado del servidor y MySql como servidor de base de datos.

Para una mejor comprensión del presente documento, la persona interesada deberá tener los conocimientos necesarios sobre las siguientes tecnologías:

- Base de datos
- Arquitectura WEB
- Lenguaje de programación PHP
- Lenguaje de programación Javascript

2. Estandarización del diseño de la Base de Datos

En las siguientes líneas se desea dar las pautas que normalicen el diseño e implementación de la base de datos.

2.1 Nombre de los objetos de base de datos

Para el desarrollo de este proyecto se utilizaron los siguientes tipos de objetos de base de datos:

- Tablas
- Triggers o Disparadores

Para la nomenclatura de las tablas de la base de datos, se utilizaron nombres referenciales al contenido de las mismas, unidos por un guión bajo (_) en caso de expresar nombres de más de una palabra.

A continuación se presenta algunos ejemplos de nombres de tablas:

Contenido	**Tipo de objeto**	**Nombre del objeto**
Registros de datos de clientes	Tabla	CLIENTES
Registros de despacho de productos a distribuidor	Tabla	DESPACHO_DISTRIBUIDOR
Registros de cantidad de stock de insumos	Tabla	STOCK_INSUMOS
Registros de devolución de envases retornables	Tabla	REGISTRO_DEVOLUCION_ENVASES

Tabla 1. Ejemplo de nomenclatura de tablas de Base de Datos
Fuente: propia

Para la nomenclatura de los triggers o disparadores de la base de datos, se utilizaron nombres referenciales a la acción que cumplen los mismos, empezando con un verbo que indica la acción a ejecutar unidos por el o los nombres de la entidad a ser afectada, los mismos que empiezan con mayúscula la primera letra.

A continuación se presenta algunos ejemplos de nombres de los triggers o disparadores:

Acción	Tipo de objeto	Nombre del objeto
Actualizar el stock de los insumos.	Trigger	actualizarStockInsumos
Ingresar en la tabla de stock productos los registros al momento de crear un nuevo producto.	Trigger	ingresoStockProductos
Actualizar el registro de devolución de envases retornables.	Trigger	actualizarDevolucionEnvases

Tabla 2. Ejemplo de nomenclatura de triggers de Base de Datos
Fuente: propia

3. Estandarización del lenguaje de programación

Ext JS utiliza a Javascript como lenguaje de programación y este a su vez utiliza la notación JSON para la definición de los objetos.

La programación de objetos y/o componentes en Ext JS se la puede realizar de varias maneras, la única condición es la utilización de los objetos de configuración que no son más que una forma de estructurar datos, los mismos que pueden ser fácilmente interpretados por los lenguajes de programación en este caso Javascript.

Para la implementación de los componentes utilizados en el sistema, se dividió la programación en tres partes:

- Diseño de formularios
- Programación de acciones de formulario
- Programación de Stores

La programación de los componentes se la realiza en archivos separados con la extensión ".js", es decir cada componente programado deberá constar en un archivo por separado nombrado con la misma nomenclatura que la definición del objeto de programación de acciones de formulario que veremos más adelante.

Es así que la programación de componentes con Ext Js se la realiza de la siguiente manera:

3.1 Diseño de formularios

Para el diseño de los formularios se creará un objeto, el mismo que extiende la definición de la clase Ext.form.FormPanel de la siguiente manera:

```
WinRegistrarProductoUi = Ext.extend(Ext.form.FormPanel, {

// Código del objeto

});
```

Los nombres de los objetos empezarán con el prefijo Win, seguido por un nombre iniciado con mayúsculas en su primer carácter referente al contenido u acción que va a ejecutar el formulario y terminado por las siglas Ui que significan User Interface (Interfaz de Usuario).

Prefijo	Referencia a la acción destinada	Terminación
Win	RegistrarProducto	Ui
Win	RegistrarCobros	Ui
Win	ReporteProduccion	Ui

Tabla 3. Ejemplo de nomenclatura de objetos de diseño de formularios
Fuente: propia

3.2 Programación de acciones de formularios

Para programar las acciones de los formularios se creará un objeto, el mismo que extiende la definición del objeto de diseño de formulario de la siguiente manera:

```
WinRegistrarProducto = Ext.extend(WinRegistrarProductoUi, {
    initComponent: function( ) {
        WinRegistrarProducto.superclass.initComponent.call(this);
                //Código de las acciones
    }
});
```

Los nombres de los objetos empezaran con el prefijo Win, seguido por un nombre iniciado con mayúsculas en su primer carácter referente al contenido u acción que va a ejecutar el formulario el mismo que tiene que coincidir con el nombre del objeto de diseño del cual se heredan los componentes.

Prefijo	Referencia a la acción destinada
Win	RegistrarProducto
Win	RegistrarCobros
Win	ReporteProduccion

Tabla 4. Ejemplo de nomenclatura de objetos de programación de acciones de formularios
Fuente: propia

3.3 Programación de stores

Los stores son almacenes temporales de datos, los cuales sirven para la comunicación e intercambio de datos de Javascript con el lenguaje del servidor, la programación de los mismos se la realiza creando un objeto el cual extiende la definición de la clase Ext.data.JsonStore de la siguiente manera:

```
clienteStore = Ext.extend(Ext.data.JsonStore, {
    constructor: function(cfg) {
        cfg = cfg || {};
        clienteStore.superclass.constructor.call(this, Ext.apply({
                    //configuración del store
         }, cfg));
    }
});
```

Los nombres de los objetos empezaran con el nombre de la entidad que va a interactuar con la base de datos, seguido por la palabra “Store”.

Nomenclatura	Descripción
clienteStore	Proveerá un almacén de datos de la información de los clientes.
insumoStore	Proveerá un almacén de datos de la información de los insumos.
produccionStore	Proveerá un almacén de datos de la información de los registros de producción.

Tabla 5. Ejemplo de nomenclatura de objetos de programación de stores
Fuente: propia

3.4 Nombres de funciones

Para la nomenclatura de las funciones que se crearan en caso de ser necesario se deben identificar mediante mínimo dos palabras que hagan referencia al objetivo de la función, por ejemplo:

- GuardarRegistro( )
- CalcularTotales( )
- LimpiarCampos( )

3.5 Documentación y comentarios en el código

Todo archivo de código tendrá como encabezado las siguientes líneas:

/*

Creado por: Danilo Vásquez

Fecha de creación: 24/09/2011

Última modificación: 24/09/2011

Descripción del archivo: Una breve descripción sobre el contenido del archivo

*/