Universidad de las Ciencias Informáticas
Facultad 4

Título: Implementación de las validaciones en JavaScript para la AGR. Adaptación al framework ExtJS

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor(es): Yandier Mayo Leyva

Tutor(es): Ing. Manuel Ramón Almaguer Ochoa

Ciudad de la Habana, Cuba

Junio, 2008

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Yandier Mayo Leyva	Manuel Ramón Almaguer Ochoa
_____	_____
Firma del Autor	Firma del Tutor

DATOS DE CONTACTO

Tutor:
Ing. Manuel Ramón Almaguer Ochoa Ingeniero en Ciencias Informáticas.
Graduado en el 2007 en la Universidad de Ciencias Informáticas (UCI).
Adiestrado en la Facultad 4 en la UCI.
Jefe de Proyecto en el Polo Productivo Sistemas.
Tributarios y de Aduana.
Correo Electrónico: malmaguer@uci.cu

AGRADECIMIENTOS

Agradezco a todas las personas que de una forma u otra han contribuido en mi desarrollo profesional, que cada día han estado ahí para enseñarme nuevas cosas de la vida y llegar a la meta que me he propuesto. Esas personas que han sabido extender su mano para ayudarme a salir del mal momento en que a veces me he encontrado, que con su comprensión y amistad han hecho todo lo posible para poder salir adelante en todos los pasos de mi carrera. Gracias por la paciencia de todos aquellos que han estado a mi lado cada segundo de preocupación y dedicación, muchas gracias. A mis amigos por los consejos, por la seguridad que siempre han tenido en mí.

Gracias a esas personas que aunque están lejos se han preocupado, mostrando su apoyo incondicional sabiendo que estamos en sus corazones. Gracias a mi madre, a mi familia, esa que me apoya, que llena de felicidad en los pasos difíciles que tenemos que dar en la vida. Y gracias a la vida por hacer que todas estas personas hagan valer nuestra existencia en el transcurso de ella.

Quiero dar las gracias a Jesucristo que me da sabiduría, fuerzas y fe para no rendirme y seguir hacia adelante, porque sin él no somos nadie.

DEDICATORIA

A mi madre, por ser la guía de mis sueños e impulsora de todas mis decisiones.
A mi abuelita, por la confianza que siempre te ha caracterizado.
A mis tías, por su preocupación y dedicación en todos estos años de estudio.
A mi hermano, que tanto quiero y estimo sobre todas las cosas.
A mi tutor Manuel, por su apoyo incondicional y confianza.
A mi gran amiga Yanet, por todo el amor dedicado en mí desde el principio de esta bella amistad.
En especial a mis amigos Trinchet, Islandy, Cesar, Mandy que en momentos de ayuda siempre conté con su apoyo incondicional.
A mis amigos “todos” por el apoyo incondicional a mi formación como profesional.
Sin todos ustedes, esto hubiera sido imposible, sinceramente Gracias.

RESUMEN

El desarrollo de la informática en todo el mundo y en especial en el país ha conllevado que la informatización se lleve a cabo en todas las esferas de la sociedad para lograr los procesos fundamentales de las instituciones la informatización de estos, logrando mayor rapidez, control y confiabilidad del manejo de la información. Hoy contamos con dos paradigmas, el Software propietario y el Software libre, el país está optando por migrar hacia el Software libre, en esta misión la universidad abanderada, dentro de los proyectos que se realizan está el proyecto Sistema Único de Aduanas (SUA).

En la Aduana General de la República (AGR) en Cuba se desarrolla una mejor informatización de los procesos, a los cuales le han asignado los recursos necesarios para un mejor desempeño de esos procesos. Por esta razón se requiere de un sistema que cumpla con las exigencias de los inspectores de las aduanas para poder brindar una respuesta en un tiempo óptimo, y de información confiable, con la informatización de los principales procesos en especial que coordinen con las nuevas tecnologías y tendencias empleadas para la informatización de esta.

El desarrollo del presente trabajo propone las validaciones de datos que se realizan del lado del cliente de una manera más organizada y segura para el usuario final, como para los propios desarrolladores del sistema. Aunque actualmente existen numerosas propuestas de frameworks para distintos dominios de aplicación. El framework JavaScript a utilizar es el ExtJS para adaptar las funciones de validación, proporcionando una infraestructura genérica que facilita la especificación y uso de las nuevas técnicas de programación web. El ExtJS se utiliza para mejorar sus funcionalidades en la validación de datos de manera más ágil del lado del cliente para la AGR.

PALABRAS CLAVE

Framework, librería, JavaScript, funcionalidades.

TABLA DE CONTENIDOS

INTRODUCCIÓN

Dado el creciente auge de las empresas cubanas en el desarrollo de nuevas tecnologías y la producción de software, se pretende ilustrar en este trabajo una alternativa de utilización de tecnologías y herramientas que sean más factibles para el desarrollo de sistemas web haciendo uso del framework Symfony y la librería ExtJS, con el objetivo de maximizar y economizar dicha producción. Con esta investigación, se dará solución a una de las problemática existente en el proyecto Sistema Único Aduana (SUA).

El desarrollo de los frameworks es de gran importancia para construir grandes sistemas de software orientados a objetos con el objetivo de maximizar la productividad, reutilización tanto del diseño como de las implementaciones y economizar tiempo de trabajo de programadores. La utilización de un framework en el desarrollo de una aplicación o sistema implica cierto grado de aprendizaje, aunque a largo plazo puede facilitar tanto el desarrollo como el mantenimiento. Por lo que es fácil encontrar herramientas (utilidades, librerías) adaptadas al framework concreto para facilitar y agilizar el proceso de desarrollo de aplicaciones web. Permite reutilizar código ya existente y promueve buenas prácticas de desarrollo.

En la Aduana General de la República (AGR) existe la necesidad de realizar un conjunto de validaciones específicas de la Institución. En el Sistema Único de Aduana (SUA) hoy día se realizan validaciones, que se encuentran de manera desorganizada, a pesar de intentos para organizar el código resulta muy difícil. El principal problema que se desea solucionar es como integrar todas las validaciones que se requieren en el SUA a la librería JavaScript que se designe para utilizar. Para posteriormente junto con las validaciones que realiza el framework que se utiliza en el proyecto hacer más segura y organizada posible la capa presentación.

Se determinó la utilización del framework de PHP Symfony para la integración de una librería JavaScript a utilizar en el momento de realizar un mejor empleo de esta facilidades que brindan para la Aduana General de la República (AGR), como ejemplo se puede mencionar una de ellas, ExtJS, que es a la cual va enfocada este proyecto investigativo.

La utilización de librerías y herramientas de JavaScript orientadas del lado del cliente se han ganado su posición en esta rama de la Informática, por lo que se puede puntualizar que luego de un estudio profundo de la situación antes planteada se logró identificar el siguiente **problema:**

Cómo adaptar al framework ExtJS las validaciones necesarias del SUA.

Objeto de estudio: La validación de los datos en el Proyecto SUA.

El **campo de acción** de este trabajo es la validación de los datos mediante el lenguaje de programación JavaScript.

Para guiar la investigación se planteó la siguiente **hipótesis:** Si se implementan las validaciones necesarias para el proyecto SUA mediante el lenguaje de programación JavaScript, y que a su vez se integren al ExtJS, se reduce la entrada de datos erróneos en el sistema, se ahorran peticiones innecesarias al servidor web, logrando una mayor robustez.

El **objetivo general** de la investigación es proponer una solución que permita evitar la entrada de datos erróneos en la aplicación mediante las implementaciones de las validaciones necesarias, integrándoselas a al framework JavaScript ExtJS.

Para lograr dichos objetivos se plantaron las siguientes **tareas:**

- Comparación entre las diferentes librerías JavaScript.
- Investigar acerca de la librería JavaScript a utilizar por el proyecto.
- Integración del ExtJS al Symfony.
- Definir que validaciones realiza symfony en la capa de presentación.
- Definir que validaciones realiza la librería ExtJS.
- Estudio de todas las validaciones que requiere el SUA.
- Implementación y prueba de las validaciones.

Este documento esta estructurado en tres capítulos:

Capítulo 1. **Fundamentación Teórica:** Se estudia las librerías que existen, vinculados al campo de acción, además de hacer una investigación de tecnologías, metodologías y herramientas actuales que se utilizarán para el desarrollo de las implementaciones que se requieren para el proyecto.

Capítulo 2. **Argumentación de la Propuesta:** En este capítulo se dará una explicación bien detallada del framework Symfony, a emplear en el proyecto y la librería JavaScript ExtJS con sus aportes a las validaciones presentes en ella.

Capítulo 3. **Desarrollo de Validaciones:** Se desarrollará la integración de la librería ExtJS con el framework Symfony. El paso final en el completamiento de los objetivos de la investigación es la implementación de las validaciones necesarias para el SUA adaptadas al ExtJS.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

Como ha demostrado la historia, las tecnologías y las tendencias que existen desde que surgió el hombre, ya que en busca de satisfacer las necesidades que estos tenían, el hombre se vio en la obligación de crear técnicas y herramientas que le permitieran resolver los diferentes problemas que se iban presentando. Al pasar de los años el hombre fue perfeccionando sus métodos y técnicas, logrando un paso en la evolución de las tecnologías. Como resultado de una evolución continua y progresiva se fueron desarrollando poco a poco lo que hoy se llama técnicas y tendencias actuales que han permitido desarrollar el mundo de la informática, con estas también han surgido herramientas y metodologías. Se puede decir que las metodologías se crearon a partir de la recopilación de varias ideas que existían de cómo se podía estructurar el desarrollo de un software.

Las exigencias para la solución de problemas cada vez más complejos, resultan muy rigurosos e introducen premisas ineludibles al desarrollo de software:

- Complejidad: Los sistemas cada vez son más y más complejos, lo que presupone que “a mas líneas de código mas errores”.
- Extensibilidad: Los sistemas deben permitir incorporar funcionalidades sin necesidad de recompilar la totalidad del código.
- Conectividad: Los sistemas deben estar preparados para interactuar con otros sistemas.
- Reusabilidad: Los sistemas deben tener la posibilidad de reutilizar software de otros proyectos o desde partes del mismo proyecto.

Después de un largo proceso de desarrollo se puede decir que todas estas herramientas, tecnologías, metodologías es lo que conforma la estructura y la garantía del desarrollo de un sistema informático seguro.

En la actualidad el mundo se debate principalmente entre dos paradigmas de desarrollo, el software propietario y el software libre. Cuba en medio de un proceso de informatización de la sociedad opta por lograr la migración completa al segundo mencionado, que permite acceder al código fuente haciéndole los cambios que requiera el usuario, sean cuales quiera sus intenciones con el mismo, con la única condición de redistribuir el software libremente licenciado bajo algún tipo de licencia de software libre que beneficie a la comunidad. Una vez que un producto de software libre ha empezado a circular, rápidamente está disponible a un costo muy bajo o sin costo alguno. Al mismo tiempo, su utilidad no decrece. Esto significa que el software libre se puede caracterizar como un bien público en lugar de un

bien privado. La mayoría del software libre se produce por equipos internacionales que cooperan a través de la libre asociación. Los equipos están típicamente compuestos por individuos con una amplia variedad de motivaciones, que permite ampliar las soluciones a problemas de la vida diaria, que serán solucionados por quienes los sufren. Esto abre una puerta a la teoría de la masividad del conocimiento, todo el que tenga acceso al código fuente con un poco de estudio puede satisfacer sus necesidades.
En este capítulo se analizan aspectos de carácter teórico, que es sumamente necesario investigar puesto que es importante tener los argumentos para hacer una propuesta que cumpla con las necesidades planteadas y de una solución lo mas eficiente posible para el problema. De aquí el estudio de algunas librerías existentes vinculadas al campo de acción del problema propuesto y señalar brevemente los diferentes conceptos y características relacionados con las tendencias, tecnologías y metodologías actuales. Para a partir de una investigación de estas, determinar cuáles son las más indicadas con el objetivo de emplearlas en el cumplimiento de varias de las tareas a cumplir.

1.2 Framework

Muchos de los que se dedican al desarrollo de software utilizan, conocen o como mínimo, se han tropezado con el concepto de framework, cuya traducción aproximada seria “marco de trabajo”. Sin embargo, el concepto de framework no es tan simple y mucho menos sencillo de definir, aunque cualquiera con experiencia en la programación captara su sentido de manera casi intuitiva e incluso es posible que este utilizando su propio framework. Resulta pertinente citar las siguientes definiciones:

- “… diseño abstracto orientado a objetos para un determinado tipo de aplicación, que se compone de una clase abstracta para cada componente principal del diseño; contendrá normalmente una librería de subclases que pueden ser utilizadas como componentes del diseño…” (1)
- “… patrón arquitectónico que proporciona una plantilla extensible para aplicaciones dentro de un dominio…” (2)

Un framework esta asociado a un determinado tipo de aplicación, lo que implica que su alcance este acotado; no es un diseño de propósito general, sino que esta ligado a un dominio concreto. En ese sentido, se usa también la terminología “plantilla extensible” para dar a entender, por un lado, que se compone de una estructura común para un determinado tipo de aplicaciones que pueden ser extendidas con nuevos componentes.
Un framework incorpora una o varias APIs que describen como tener acceso a las diferentes bibliotecas de clases que presentan que representan los puntos calientes (hot spots) de dicho

framework. A este conjunto de clases relacionadas y reutilizables, diseñadas para proporcionar funcionalidades útiles de propósito general se le conoce como ToolKit. **(3)** Los cuales no imponen un diseño particular en una aplicación, simplemente proporcionan funcionalidades que pueden ayudar a realizar el trabajo.

1.2.1 Ventajas de producción y mantenimiento

Los frameworks aportan ventajas a la producción y al mantenimiento debido a la flexibilidad y consistencia que brinda su empleo correcto. Al utilizar un framework de forma adecuada el software puede responder fácilmente a los cambios en los requerimientos tanto del negocio como de tecnología. El mantenimiento de las aplicaciones históricamente ha requerido descubrir cómo un desarrollador solucionó un problema determinado y luego cambiar ese comportamiento. Una solución a esto es usar miembros del equipo de desarrollo original para mantener el software, que evidentemente no hace un uso eficiente de los recursos. En cambio al usar un framework, una solución consistente es usada a través del desarrollo y por ende lleva consigo uno de los principios fundamentales de los frameworks, la flexibilidad. Es muy fácil localizar la parte de la aplicación afectada por los nuevos requerimientos y realizar los cambios de una manera controlada.

1.3 Symfony un framework en PHP

Con el surgimiento de la informática y con el paso del tiempo se fue haciendo necesario el construir grandes sistemas de software orientado a objetos. Las empresas por su parte no se excluyen de enfrentar nuevos desafíos debido a esta problemática, y como una solución a la misma surgieron los frameworks para la reutilización tanto del diseño como de las implementaciones, una mayor productividad y una disminución considerable del tiempo de salida al mercado del producto. Entre los que podemos mencionar algunos realizados en PHP, de la cual la mayoría están basados en el patrón Modelo Vista Controlador (MVC: Model-View-Controller): CakePHP, Zend Framework y Symfony. Actualmente, este último se emplea en la realización del proyecto pues su tecnología se encuentra en prácticamente en varias industrias alrededor del mundo.

Symfony es un framework diseñado para optimizar el desarrollo de aplicaciones web a través de diversas características claves. Separa las reglas de negocio de la aplicación, la lógica del servidor y las vistas de presentación. Contiene una gran variedad de herramientas y clases para conseguir acotar

el tiempo de desarrollo de aplicaciones web complejas. Adicionalmente, automatiza tareas comunes para el programador pueda enfocarse por completo en las aplicaciones. **(4)**

1.3.1 Características de Symfony

- Fácil de instalar y configurar: Ha sido probado con éxito en plataformas Windows y derivadas de Unix.
- Independiente del manejador de base de datos: Utiliza Propel, una capa de abstracción que le permite interactuar con varias bases de datos.
- Simple de usar: Al mismo tiempo lo suficientemente flexible para adaptarse a escenarios complejos.
- Basado en la premisa de "convención sobre configuración": El desarrollador solo necesita configurar aquellos aspectos sobre los cuales no hay una tendencia definida.
- Cumple con la mayoría de las mejores prácticas en diseño web y patrones de diseño.
- Utilizable en entornos empresariales: Puede adaptarse a políticas y arquitecturas ya existentes en tecnologías de información, y es lo suficientemente estable para proyectos de largo plazo.
- Código legible: Con comentarios en phpDocumentator para su fácil mantenimiento.
- Fácil de extender: Permite la integración con otras librerías.
- Incorpora herramientas que facilitan la prueba y depuración de aplicaciones: Como unidades de generación de código, pruebas del funcionamiento del framework, panel de depuración, interfaz por línea de comandos y configuración en tiempo real. (5)

1.3.2 Automatización de características de proyectos web

Adicionalmente pueden señalarse otro grupo de características que permiten la automatización de ciertas tareas relacionadas con la construcción de aplicaciones web, como las siguientes:

- Incorpora una capa de internacionalización: Posibilita la traducción de datos e interfaces, así como la localización de contenido en función de la ubicación geográfica del usuario.
- Uso de plantillas: Pueden ser elaboradas por diseñadores de páginas web que desconocen el resto de detalles técnicos del framework.
- Validación y regeneración automática de formularios: Asegura una buena calidad de los datos en la base de datos y una mejor experiencia de usuario.
- Verificación de la salida enviada por la aplicación: Ofrece una protección frente a ataques por datos corruptos.

- Manejo de memoria caché: Reduce el uso del ancho de banda y la carga en el servidor.
- Mecanismos de autenticación y credenciales: Facilitan la creación de secciones restringidas y la gestión de seguridad de usuarios.
- URLs inteligentes: Permiten que las direcciones de las páginas web sean parte de la interfaz y resulten amigables a los motores de búsqueda.
- Una gestión de listas mas amigables al usuario: Gracias a la paginación automática, ordenamiento y filtrado de resultados.
- Las interacciones usando AJAX: Fáciles de implementar gracias a los asistentes de una sola línea de código que encapsulan efectos en JavaScript compatibles con varios navegadores. (5)

1.3.3 Validación de formularios mediante Symfony.

La primera vez que se accede al código fuente de una aplicación realizada con Symfony, puede desanimar un poco a los nuevos desarrolladores. El código está dividido en muchos directorios y scripts. Los archivos son un conjunto de clases PHP, código HTML e incluso una mezcla de los dos. Además, existen referencias a clases que no se pueden encontrar dentro del directorio del proyecto y anidación de directorios puede llegar hasta los seis niveles. Sin embargo, cuando se aprenden las razones que están detrás de esta aparente complejidad, se verá como algo completamente natural y sin necesidad de cambiar la estructura de una aplicación.

Symfony está basado en un patrón clásico del diseño web conocido como arquitectura MVC **(6)**, que está formado por 3 niveles:

- El modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.
- La vista transforma el modelo en una página web que permite al usuario interactuar con ella.
- El controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones.

En la capa de la Vista se necesitan para realizar páginas sencillas para las que son necesarios los siguientes componentes:

- La capa de la Vista
    - Vista
    - Plantilla

- Layout

Symfony utiliza lo mejor de la arquitectura MVC y la implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo.

La lógica de la vista se puede transformar en un archivo de configuración sencillo, sin necesidad de programarla. Incluir formularios en las plantillas es muy sencillo gracias a los helpers de formularios que incluye Symfony y a sus opciones avanzadas. Si se definen formularios para modificar las propiedades de un objeto, los helpers de formularios para objetos simplifican enormemente su desarrollo. Los archivos de validación, los helpers de validación y la opción de volver a mostrar los datos en un formulario, permiten reducir el esfuerzo necesario para crear un control estricto de los formularios para que sea robusto y a la vez fácil de utilizar por parte de los usuarios. Además, cualquier validación por muy compleja que sea se puede realizar escribiendo un validador propio o utilizando un método validateXXX() en la clase acción.

1.3.4 Tendencias actuales

En este momento, los frameworks para el desarrollo de aplicaciones web están en pleno apogeo, y es necesario disponer de un completo framework realizado con PHP. Symfony proporciona una solución irresistible a esa carencia, debido a la calidad de su código fuente y a la gran cantidad de documentación disponible, dos ventajas muy importantes sobre otros frameworks disponibles. Los colaboradores aparecieron en seguida proponiendo parches y mejoras, detectando los errores de la documentación y realizando otras tareas muy importantes.

Si se desarrollan aplicaciones web complejas con mucha lógica de negocio, no es recomendable utilizar solo PHP. Para asegurar el mantenimiento y las ampliaciones futuras de la aplicación, es necesario que el código sea ligero, legible y efectivo. Para incorporar los últimos avances en interacción con usuarios (como por ejemplo AJAX), se puede acabar escribiendo cientos de líneas de JavaScript. Para desarrollar aplicaciones de forma divertida y muy rápida, no es aconsejable utilizar solo PHP. En todos estos casos, de utilizar un framework de desarrollo de aplicaciones web y solo se necesita uno que sea maduro, bien documentado y con una gran comunidad que lo apoye.

1.4 Librería

En todos los lenguajes de programación existen librerías de funciones que sirven para hacer cosas diversas y muy repetitivas a la hora de programar. Las librerías de los lenguajes de programación

ahorran la tarea de escribir las funciones comunes que por lo general pueden necesitar los programadores. Un lenguaje de programación bien desarrollado tendrá una buena cantidad de ellas. En ocasiones es mas complicado conocer bien todas las librerías que aprender a programar en un lenguaje. **(7)**

Dos cualidades que pueden distinguir rápidamente una librería de clases de un framework son la integridad y la extensibilidad. Una librería de clases ejecuta una tarea completa para una categoría específica de problemas; la solución queda formada cuando el usuario le añade comportamiento al framework. La extensibilidad es útil a la hora de distinguir entre los frameworks y las librerías de clases. Las librerías de clases son comúnmente usadas con un comportamiento de caja negra, no exponen el funcionamiento de las clases.

En la práctica los frameworks y las librerías de clases son tecnologías complementarias. Por ejemplo los frameworks frecuentemente usan librerías de para simplificar el desarrollo del propio framework. Asimismo el código específico de una aplicación que es invocado por los manejadores de eventos del framework pueden utilizar las librerías de clases para realizar tareas básicas como el manejo de cadenas, la administración de ficheros, análisis numéricos. JavaScript contiene una buena cantidad de funciones en sus librerías. Como se trata de un lenguaje que trabaja con objetos muchas de las librerías se implementan a través de objetos.

1.5 Lenguaje de Programación.

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.

1.5.1 JavaScript.

JavaScript es un lenguaje de programación que ha permitido el gran desarrollo de la web, ha sido el avance mas significativo en el logro de páginas web dinámicas y exactas en cuanto a la posición y presentación de su contenido. Es un lenguaje robusto y a la vez ligero, el cual a pesar de ser considerado por muchos como un lenguaje no orientado a objetos permite implementar varias de las características de este paradigma de programación basado en prototipos, las nuevas clases se generan clonando las clases bases y extendiendo su funcionalidad, cualquier navegador puede interpretar el código JavaScript dentro de la página web.

JavaScript es un lenguaje interpretado, es decir, que no requiere compilación, utilizando principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y el de C. Lenguaje de programación utilizando para crear pequeños programas encargados de realizar acciones dentro del ámbito de una página web. Se trata de un lenguaje de programación del lado del cliente compatible con la mayoría de los navegadores modernos, por consiguiente permite la máxima interactividad entre el usuario y la página, además la verificación de los datos introducidos por el usuario antes de enviar el formulario al servidor, el manejo de applets y plugins dentro de múltiples marcos de HTML. El navegador del cliente es el encargado de interpretar las instrucciones de JavaScript y ejecutarlas. JavaScript es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas.

Este lenguaje de programación tiene algunas limitaciones:

- JavaScript por definición no es un lenguaje orientado a objetos y debido a la potencia de esta programación se ha conseguido un funcionamiento similar, intuitivo y potente.
- No se compila, porque lo que hace es interpretar el código.
- No es obligatorio declarar variables pero muy importante para la utilización del ámbito de las variables globales o locales.
- Verifica las referencias en tiempo de ejecución.
- No tiene protección del código, ya que se descarga en texto claro. (8)

1.5.2 Fundamentación de la selección del lenguaje de programación web a utilizar.

Se decidió utilizar JavaScript por su compatibilidad con los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado, permite la validación del lado del cliente evitando accesos innecesarios al servidor lo que puede ser molesto tanto para el usuario como para nuestro servidor, que se carga inútilmente.

Uno de los inconvenientes más comunes a la hora de diseñar una interfaz de aplicación Web es que una vez que la página se ha descargado en el cliente, la conexión con el servidor se corta. Cualquier intento de dinamismo en la interfaz por parte del cliente requiere una comunicación con el servidor para la recarga (proceso que tiende a convertir la aplicación poco elegante y lenta).

En el modelo tradicional de aplicaciones Web el usuario envía una petición al servidor requiriendo una página, la cual es construida y enviada al navegador. Esta página incluye un formulario HTML para capturar datos del usuario. Una vez que el usuario reenvía los datos al servidor, la siguiente página

será generada y enviada dependiendo del valor de dichos datos, y así el proceso continúa. Supongamos una aplicación de escritorio para el registro de un número de serie. Según convenga se lo puede plantear de diversas formas, una vez hemos terminado de rellenar los correspondientes cuadro de textos con los caracteres del código, podríamos hacer aparecer una "Tilde" verde a la derecha indicando que se ha introducido un código válido. Tan pronto como se introduce el código, la aplicación puede comprobar su validez y responder.

En contraste con el ejemplo anterior pero esta vez orientado en una interfaz Web. Por supuesto, todos los cuadros de textos para introducir el código serán idénticos, pero al rellenarlos, el usuario tendrá que enviar esos datos al servidor para que éste valide el código. Una nueva página será entonces cargada informando del éxito o fracaso de la operación, y en caso de fallo, el usuario tendrá que volver atrás e intentarlo de nuevo cuantas veces sea necesario. Una solución a estos problemas se presenta con el objeto XMLHTTPRequest. Este objeto, ahora disponible como objeto nativo tanto en Mozilla como también en otros navegadores existentes, permite a JavaScript realizar peticiones al servidor remoto sin la necesidad de recargar la página. En esencia, pueden realizarse peticiones y recibir respuestas HTTP completamente en segundo plano y sin que el usuario experimente ninguna interrupción visual.

1.6 Descripción de Librerías

Seguidamente se abordarán aspectos de carácter teórico relacionados con las características de las diferentes librerías JavaScript para la programación web.

1.6.1 Librería Prototype

En el 2005, con la popularización de la tecnología basada en el objeto xmlHttpRequest, surgieron nuevas ideas para interfaces. Con él, surge la librería Prototype que ayuda sensiblemente a la implementación de AJAX en los sitios. Prototype resultó ser una librería muy efectiva y fácil de usar, además de integrarse independientemente del proyecto Ruby on Rails, por lo que se hizo muy popular. Prototype es una librería que marcó un paso adelante en lo que respecta al desarrollo de páginas ágiles e interactivas. **(9)**

El cambio en el mundo de la programación web, hace uso de las bondades de la web 2.0. Con este cambio las técnicas de desarrollo de páginas web necesitaban dar un gran salto. Prototype brinda su aporte a este auge, para lograr mejores servicios a menor costo de desarrollo. Es la librería más utilizada de la red, aunque tal vez no es la mejor ya que tiene competencia muy fuerte con JQuery y

MooTool, pero es muy útil. Su gran desventaja siempre ha sido el peso de la librería, pero hay opciones muy buenas para comprimirlas.
El potencial de Prototype es aprovechado al máximo si se desarrolla con Ruby On Rails, esto no quiere decir que no se puede usar desde otro lenguaje, solamente que demandará un mayor esfuerzo en el desarrollo. Prototype es una librería JavaScript que apunta al desarrollo sencillo y dinámico de aplicaciones web. Es una herramienta para el desarrollo de clases única y de fácil uso, además de ser la biblioteca más agradable de AJAX.
La parte más grande de la librería Prototype son sus extensiones de DOM. Prototype agrega muchos métodos de conveniencia. Todos los elementos de DOM tienen los métodos de extensión de Prototype incorporado. Esto, sin embargo no es verdad para IE que no permite que nadie toque HTMLElement.prototype. En IE se extiende el elemento con Element.extend(). Debido a navegadores que no apoyan esto, se debe tener cuidado para solo usar extensiones de DOM en elementos que no han estado extendidos.

1.6.1.1 Características

- Desarrollo sencillo de aplicaciones AJAX: Además de las llamadas sencillas, este modulo también proporciona una forma sencilla de trabajar con el código JavaScript enviado por el servidor. También proporciona clases para facilitar las tareas de llamadas selectivas.
- Extensión de DOM: Añade varios métodos a los elementos retornados con la función $(). Por ejemplo, puedes realizar la siguiente operación:
  $('comentarios').addClassName('activo').show() para obtener el elemento con el identificador(id) 'comentarios', cambiar el nombre de su clase y mostrarlo (si estaba oculto)
- Utilidades JSON (JavaScript Object Notation): JSON es una alternativa más ligera que las llamadas XML realizadas con AJAX. **(19)**

Licencia MIT:
La autorización se concede, de forma gratuita, a cualquier persona que obtenga una copia de este software y los archivos de documentación asociados, para trabajar con el Software sin restricción, incluyendo la limitación, los derechos para utilizar, copiar, modificar, fusionar, publicar, distribuir o vender copias del Software, y permitir a las personas a las cuales se suministra el software para hacerlo, con las siguientes condiciones:

El software se proporciona originalmente, sin garantía de ningún tipo, expresa o implícita, pero sin limitarse a las garantías de comerciabilidad, aptitud para un propósito en particular y no infracción. En ningún caso, los autores o titulares de derechos de autor serán responsables de cualquier reclamación, daños o cualquier otra responsabilidad, ya sea en una acción de contrato, agravio o de otra índole, derivadas de afuera de sus usos con el software, u otros tratos en el software. **(9)**

1.6.2. Librería Dojo Toolkit

La librería Dojo Toolkit es una librería JavaScript de código abierto DHTML (Dynamic HTML) escrito en JavaScript. Proporciona un API para el control y manipulación de historial, permite la manipulación de URL y marcadores/favoritos, el trabajo con widgets e incluye el ordenamiento de tablas, validación de formularios, menús y barra de menús, Google y Yahoo! Maps.

Su idea es la de abstraer al desarrollador de las complejidades de DHTML y de las discrepancias existentes entre navegadores, que hacen el código JavaScript a utilizar sea diferente.

Dojo Toolkit es hecho de un juego de bibliotecas de capas. También puede usar el API de bajo nivel y capas compatibles para escribir scripts portables y simplificar aquellos que son complejos que le permite personalizar la distribución de Dojo para la aplicación.

Proporciona un potente entorno de programación incorporando un sistema de control de eventos, un API E/S y un lenguaje estándar mejorado que grandemente mejoran y simplifican la vida de JavaScript para diseñadores. Se puede usar las herramientas de Dojo para construir unidades prueba para el código JavaScript y optimizar el desarrollo. **(10)**

1.6.2.1 Características

- Múltiples Puntos de Entrada: Es un concepto fundamental en el diseño de Dojo. Este término significa que los usuarios pueden empezar a utilizar Dojo al nivel que ellos deseen.
- Independencia del interpretador: Asegurar soporte para el mayor número de plataformas.
- Unifica varios Codebase: Incorpora en varios frameworks (nWidgets, Burstlib, y f(m)).
- Maneja incompatibilidades entre navegadores. **(18)**

Licencia:

Dojo se utiliza libremente para construir aplicaciones y servicios. Se puede distribuir como parte de los productos comerciales. Modificar Dojo, hacer ampliaciones de la guía, y producir componentes propios sin necesidad de dar su código o de contribuir de nuevo al proyecto. Construir el software comercial sin obtener una licencia o que incurre en ninguna otra obligación. Dojo está disponible ya sea en virtud de los términos de la licencia BSD modificada o la Licencia Libre Académica versión 2,1. Ambas licencias le otorga amplios derechos de uso y construir en ambos casos de código abierto (Open Source) con Dojo y de ajustes comerciales. **(14)**

1.6.3 Librería JQuery

JQuery es una librería de JavaScript, rápida y concisa que simplifica el trabajo con documentos HTML. No es un extenso e inflamado framework que prometa lo mejor de AJAX, ni un conjunto de innecesarias y complicadas mejoras: JQuery ha sido diseñado para cambiar la forma de escribir JavaScript. Utiliza un interesante concepto para hacer código corto y simple, tiene manejadores de eventos. Otro tema que JQuery resuelve con facilidad es el de los efectos, añade dinamismo visual a la presentación del sitio, como son añadirle funcionalidad, tanto al código como al resto de los elementos. **(11)**

Los métodos de JQuery se requieren para colocar automáticamente todos los elementos de DOM en el código, y aplican el método deseado. Se elimina la iteración en el código (en la mayoría de los casos), esta es una de las ventajas prácticas, cotidianas y superiores a usar en JQuery. Dada la madurez que ha adquirido esta librería, es más fácil construir plugins a partir de una estructura ya existente, permitiendo así que se elimine prácticamente toda la iteración molesta.

Normalmente, cuando se trabaja con JavaScript el código no corre hasta tanto no se hayan cargado las imágenes, incluyendo los banners, para solucionar este problema JQuery ha implementado un procedimiento que se puede utilizar, conociendo como ready.event, este código chequea el documento y espera a que este listo para manipularlo.

También se le añaden plugins que le da versatilidad. Tiene un núcleo muy pequeño de solo 15KB, que es muy estable. JQuery y virtualmente todos sus plugins están encogidos dentro del namespace del JQuery. Como una regla general, se guardan los objetos globales también dentro del namespace, para lograr un mejor integración entre JQuery y cualquier otra biblioteca (Prototype, MooTool, o YUI).

1.6.3.1 Características

- JQuery Core: Principales funciones de JQuery ($, añadir plugins...).
- Selectores: Los selectores son una combinación de CSS 1-3. XPath, junto con funciones de código especiales que permiten que funcionen conjuntamente. De forma realmente sencilla podemos seleccionar diferentes elementos del árbol DOM mediante XPath. Ejemplo, la función JQuery (‘input[@name=email]’) selecciona el campo de texto con el nombre igual a “email”.
- Atributos: Permiten seleccionar, cambiar, leer, añadir y borrar los atributos de los elementos DOM de forma sencilla.
- Traversing: Funciones de filtrado y de búsquedas.
- Manipulación: Funciones de inserción y modificación.
- CSS: Funciones para el cambio de estilo.
- Eventos: Control de eventos para elementos DOM con la sintaxis propia de CSS, ha puesto en su web una lista de plugins y demos muy útiles.
- Efectos: Sencillos efectos visuales. Sino fuera por los plugins que hay desarrollados para JQuery quedaría muy por debajo de otros frameworks en este punto.
- Utilidades: Otras funciones genéricas adicionales, permiten referenciar objetos sin problemas de espacios de nombres. **(18)**

Licencia GPL y MIT:

JQuery está disponible para su uso personal en todos los proyectos comerciales o en virtud de ambos, así como la licencia MIT y GPL. Para poder elegir la licencia que mejor se adapte al proyecto, y en consecuencia utilizarlo.

GPL

Licencia GPL de software libre legalmente concede los permisos para ejecutar el programa con cualquier fin, estudiar como funciona el programa y adaptarlo a las propias necesidades, hacer copias del programa y distribuirlas, mejorar el programa y difundir la versión mejorada públicamente.

Para cumplir todo esto, es necesario que el código del programa sea accesible, abierto. Además, si el programa original está bajo la licencia GPL, cualquier programa derivado tiene que estarlo también. Así, cualquier mejora que se haga sobre un programa beneficia a la comunidad de la que salió originalmente.

En definitiva, la licencia GPL evita que alguien se aproveche del software libre, fruto de la colaboración entre muchas personas, sin dar nada a cambio.

MIT

La autorización se concede, de forma gratuita, a cualquier persona que obtenga una copia de este software y los archivos de documentación asociados, para trabajar con el Software sin restricción, incluyendo, sin limitación, los derechos para utilizar, copiar, modificar, fusionar, publicar, distribuir o vender copias del Software, y permitir a las personas a las cuales se suministra el software para hacerlo. **(15)**

1.6.4 Librería Interfaz de Usuario de Yahoo (YUI)

En 2006, Yahoo! rediseña su portal y decide utilizar AJAX como centro de su interfaz y decidieron crear su propio framework de JavaScript. Yahoo! User Interface (YUI) surge en ese mismo año con la finalidad de crear Interfaces de Usuario. Cuenta con una serie de utilidades y controles desarrollados en JavaScript, para la construcción de aplicaciones web interactivas y opulentas, utilizando técnicas como DOM, DHTML y AJAX, también incluye diversos recursos basados en CSS. Todos los componentes de YUI son libres para el uso de todos los que accedan a ellos, esta librería esta patentada con la licencia BSD de software libre. **(12)**

1.6.4.1 Características

Están disponibles dos tipos de componentes diferentes: Utilidades y Controles.

Las Utilidades de YUI simplifican el desarrollo para la compatibilidad entre navegadores basados en técnicas DOM, DHTML y AJAX.

Los Controles de YUI proporcionan elementos visuales altamente interactivos del diseño para sus aplicaciones web. Estos elementos se crean y se manejan íntegramente del lado del cliente (usuario) y nunca requieren de una recarga de página.

Utilidades disponibles:

Básicamente, YUIL contiene las siguientes utilidades:

- Animación: Para crear efectos de movimiento, animando la posición, el tamaño, la opacidad u otras características de los elementos de la página.

- Conector Principal: Esta librería se utiliza para el manejo de las transacciones XMLHttpRequest (AJAX).
- DOM: Abarca una variedad de métodos que simplifican el desarrollo de DOM-Scripting. Incluye la colocación de elementos y estilos CSS.
- Drag and Drop: Crea objetos que se pueden arrastrar y soltar en cualquier parte de la página.
- Event: Es una clase que permite el fácil y seguro acceso a los eventos del navegador (tales como los tecleos).

Controles disponibles:

- También dispone algunos controles muy usados:
- Calendario: Es un control gráfico y dinámico, usado para la selección de la fecha.
- Slider: Este control proporciona el elemento que facilita al usuario elegir un valor numérico.
- TreeView: Produce un menú estilo árbol, donde sus nodos se puedan ampliar y contraer, cargar dinámicamente y modificar con CSS. **(19)**

Licencia BSD:

BSD

La redistribución y el uso de este software de fuente y en forma binaria, con o sin modificaciones, siempre que se cumplan las siguientes condiciones:

Las redistribuciones del código fuente deben conservar el aviso de copyright anterior, presentes en la lista de condiciones.

En forma binaria deben reproducir el aviso de copyright, en la lista de condiciones y la siguiente renuncia en la documentación y / o otros materiales suministrados en la distribución.

Todos los YUI contenido es autor de Yahoo! y de su propiedad intelectual. YUI es expedido por Yahoo! bajo la licencia BSD. En algunos casos concretos, YUI incorporará labor realizada por los desarrolladores fuera de Yahoo! con su permiso expreso. **(16)**

1.6.5 Librería ExtJS

La programación con la librería ExtJS, que extiende la librería YUI e integra AJAX, Prototype y Scriptaculous. Con el tiempo se convirtió en un framework independiente y a principio de 2007 se creó una compañía para comercializar y dar soporte al ExtJS.

ExtJS es neutral al lenguaje que se use en el servidor. Siempre que el resultado se envíe a la página en el formato adecuado, ExtJS no se preocupará de lo que pase en el servidor. Hay docenas de widgets a escoger en ExtJS, incluyendo composiciones automáticas de páginas, pestañas, menús, barras de herramientas, diálogos, vistas en árbol. Proporciona un selector de nodos DOM extremadamente poderoso llamado DomQuery (puede usarse como una librería independiente, pero en el contexto de ExtJS se usará para seleccionar elementos para poder interactuar con ellos a través de la interfaz Element, contiene mucho de los métodos y propiedades de DOM que se necesitará proporcionando una interfaz conveniente, unificada y multinavegador). **(13)**

Para trabajar con las librerías de ExtJS es necesario que exista un adaptador, donde esta clase sobre la que luego se define las funciones de las librería (el elemento Ext como tal). A partir de la versión 1.1, ExtJS proporciona su propio adaptador, ya no es necesario incluir el de YUI, o JQuery o Scriptaculous (incluir en el caso de usar alguna función de estas librerías que no esté en Ext)

En su última versión Ext 2.0 API es muy extenso y recordando todas las funciones, propiedades o configuraciones disponible es casi imposible. La documentación del API está muy completa. Se emplean IDEs de desarrollo de JavaScript, el Aptana Studio es una herramienta potente y disponible para el apoyo directo en las aplicaciones desarrolladas en ExtJS.

1.6.5.1 Características

- Ext.Element: Representa un elemento del árbol DOM. Muchas de las funciones de manipulación de los elementos tienen un parámetro opcional que permite realizar el cambio mediante un efecto de animación. El parámetro de animación puede ser un dato booleano o un objeto que incluye las opciones de la animación.
- Ext.BorderLayout: Esta clase representa un diseño común para ser usado en aplicaciones de escritorio.
- Ext.DomHelper: Utilidades para trabajar plantillas o DOM. Soporta el uso de DOM o fragmentos de HTML de forma transparente.
- Ext.TabPanel: Un ligero contenedor de tabs.
- Ext.UpdateManager: Proporciona soporte para actualización AJAX de los objetos Element. **(19)**

Licencia:

De esta forma ExtJS tiene dos tipos de licencias, LGPL y comercial. Básicamente, se puede usar para nuestros desarrollos, pero para obtener soporte se debe tener una licencia comercial.

Licencia comercial

Muchos usuarios prefieren una licencia comercial por muchas razones. Esta licencia es apropiada para las organizaciones que:

- No quieren tener ninguna restricción potencial de una licencia de código abierto (open source).
- Tienen que tener una licencia comercial para satisfacer requerimientos de licencias en el software interno.
- Financiar el proyecto ExtJS para asegurar que este continúe con éxito.

Licencia Open Source

Ext está disponible bajo la licencia Open Source LGPL 3.0. Esta licencia es apropiada para usuarios que:

- Quieren usar ExtJS en proyectos de código abierto (open source) que descartan usar software que no utilicen este tipo de código.
- Planean usar ExtJS de forma personal, educacional y sin fines comerciales.
- Están usando ExtJS en una aplicación comercial que no es un software de desarrollo de librerías o toolkit, conociendo los requerimientos LGLPL y no desean mantener el proyecto.

Licencia para re-vendedores

La distribución de ExtJS en un producto que será empaquetado o vendido como un software de desarrollo de librerías, toolkit o framework basado en librerías, se requiere de un trabajo con el equipo de Ext para establecer una licencia específica apropiada. Usar los términos de la licencia código abierto (open source) en una librería no está permitido sin autorización explícita. Hay muchos beneficios de asociarse con ExtJS:

- Con una licencia OEM, los clientes (desarrolladores) no necesitarán conocer los términos de la licencia LGPL y podrán usar la funcionalidad de Ext bajo los términos de la licencia.
- Ofrecer el único producto basado en el ExtJS oficial en el mercado. (17)

La librería es muy grande en tamaño (aprox. 450Kb comprimida), aunque se puede comprimir con el módulo mod_deflate de Apache (utilizando gzip) además de que en el sitio de ExtJS se puede recortar la librería para incluir solamente aquellas funcionalidades que en realidad se van a utilizar.

1.7 Fundamentación de la librería JavaScript a utilizar

En esta sección se han abordado la mayoría de las librerías conocidas y más utilizadas, que de una manera u otra utilizan JavaScript para realizar sus funcionalidades más rápidamente y con mayor eficacia. Se ha intentado darle a conocer al lector las características principales y fundamentales de cada una de ellas, para que de un cierto modo seleccionar cual de ellas se adecua más a sus necesidades y conozca sus ventajas y desventajas con el fin de darle una utilización más eficiente y productiva para el Sistema Único de Aduanas. Seleccionándose para el desarrollo de las validaciones en el sistema la librería ExtJS, por sus facilidades de codificación, código reutilizable, las facilidades que brinda para la integración con otras librerías, independiente o adaptable. Una de sus grandes ventajas es que el su API es homogenizado independientemente del adaptador usado. Los controles siempre se verán igual. Con una extensa comunidad de usuarios.

En Cuba se utilizan muchas de estas librerías para un mayor desempeño de aplicaciones Web, de carácter informativo, educativo, productivo y empresarial, permitiendo de forma bastante acelerada que todas las esferas del gobierno estén a la altura del desarrollo del mundo así como la incursión de la nación en el ciberespacio. La Universidad de las Ciencias Informáticas, pionera de la producción de software, opta por lograr la migración completa hacia Software Libre, abriéndose camino en la web, recurriendo a las últimas técnicas de programación que posibilitan no solo la exportación e importación de los productos de software con mayor calidad, sino que hacen más duradera la aplicación desarrollada. Mejorando también la calidad de vida para los estudiantes vinculados al trabajo profesional. Reflejado directamente en los resultados que obtiene la universidad y la institución.

1.8 Herramienta de la programación Web

Permite el uso de técnicas y herramientas conocidas para crear controles y contenido web. Estas herramientas incluyen Visual Basic, Visual C++...

Aptana

Aptana IDE es un entorno de desarrollo dirigido hacia las aplicaciones web escritas en AJAX/JavaScript, multiplataforma y desde hace poco tiempo de código abierto enfocado al desarrollo de interfaces de usuario web dinámicas, ideal para maquetar páginas web. **(20)** Este software se integra en Eclipse, el famoso entorno de desarrollo Java (Lenguaje en el que esta escrito Aptana), nos permite editar fácilmente HTML, CSS y JavaScript. Las características de este IDE son similares a otros IDE más generales: Gestión de Proyectos, Vista en Línea y Vista Previa, Auto Completado, Marcos (en este caso escritos en JavaScript), Gestión de Documentación. Soporta las librerías más

populares: Prototype, Scriptaculous, Dojo, Mochikit, Yahoo! UI, ExtJS, Aflax, JQuery y Rico, pudiendo combinarlas fácilmente en la aplicación. Incorpora información sobre la compatibilidad entre navegadores, el de JavaScript asiste sobre las funciones variables específicas del lenguaje y también las definidas en el script. Incluye un explorador de archivos y de proyectos que nos permitan cambiar fácilmente entre los archivos que tenemos abiertos y nos da acceso a la mayoría de funciones y plugins de Aptana desde la misma ventana. **(21)**

1.9 Navegador web

Aplicación que permite visualizar la información que contienen las páginas en HTML. Es el programa que nos ofrece acceso a Internet. Debe ser capaz de comunicarse con un servidor y comprender el lenguaje de todas las herramientas que manejan la información de la web. Puede decirse que cada tipo de software podría tener su navegador propio, aunque los más populares sean Internet Explorer y Mozilla Firefox.

Mozilla Firefox

El navegador sobre el cual correrá la aplicación es el Mozilla Firefox este permite abrir por defecto las nuevas páginas web en pestañas. Cada una de esas pestañas tiene su propio botón de cerrado. Otras de las ventajas es restaurar sesión por ejemplo, si el Firefox tiene que reiniciarse o cerrarse, cuando se inicie de nuevo estará exactamente como se dejo abierto, no permitiendo así la dirección de la web a donde estas trabajando. Posee un corrector ortográfico integrado, en caso que se cometan errores de ortografía en las entradas de información que se haga. Tiene una sugerencia de búsqueda que se va desplazando a medida que vas introduciendo el texto por lo que se quiere buscar. El bloqueador de Firefox avisa cuando se bloquean ventanas emergentes mediante una barra informativa o un icono en la parte inferior de la pantalla. También mantiene a salvo de programas espías e impostores, usando el poder de una comunidad apasionada que te protege 24 horas al día, 7 días a la semana. La protección antiphishing lleva la seguridad de Firefox a un nuevo nivel, cuando se encuentra una página web que sea sospechosa de fraude el Mozilla Firefox advierte y ofrece una página de búsqueda para encontrar la página web que realmente se estaba buscando. Estas y muchas otras posibilidades brindan este navegador del cual se hará uso para el sistema. **(22)**

1.10 Conclusiones

Se puede concluir que a partir de un estudio hecho en la Aduana General de la República se demostró que la validación de datos en el sistema es de vital importancia para el control de los mismos. Por esta razón se propuso el desarrollo de un framework que contenga en su realización la validación de los datos. Después de un análisis de las herramientas existentes que se podían utilizar y teniendo en cuenta las políticas para el desarrollo de aplicaciones y de haber estudiado todas las ventajas y desventajas, para el sistema propuesto se escogieron entra las herramientas para facilitar el desarrollo, el Aptana Studio debido a las facilidades de uso para la programación en el lenguaje JavaScript y dentro de su amplias librerías se escogió ExtJS para implementaciones de validaciones del lado del cliente y como el navegador Mozilla Firefox, para el desarrollo del software.

CAPÍTULO 2: ARGUMENTACIÓN DE LA PROPUESTA

2.1 Introducción

Symfony, es un framework de desarrollo para aplicaciones web. Simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes, además proporciona estructura al código fuente, forzando al desarrollador a crear código más legible, más fácil de mantener y facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas.

ExtJS es simplemente un framework escrito en JavaScript que permite desarrollar aplicaciones web complejas de una forma cómoda sin tener que lidiar con los típicos problemas de la programación. Soporte para construir interfaces gráficas complejas y dinámicas, comunicar datos de forma asíncrona con el servidor y manejar datos de distinta índole de una manera simple. ExtJS se ha basado en otros framewroks previamente construidos y los han unificado obteniendo una mayor potencialidad.

2.2 Composición de Symfony

Los plugins

Los plugins permiten agrupar todo el código diseminado por diferentes archivos y reutilizar este código en otros proyectos, encapsular clases, helpers, módulos, esquemas y extensiones para el modelo, etc. Básicamente, un plugins es una extensión encapsulada para un proyecto Symfony.

Validadores, Filtros

Los filtros son trozos de código ejecutados para cada petición, antes o después de una acción. Por ejemplo, los filtros de seguridad y validación son comúnmente utilizados en aplicaciones web. La validación de los datos de la acción (normalmente los parámetros de la petición). Symfony incluye un sistema de validación, utilizando métodos de la clase acción.

Plantillas

La vista se encarga de producir las páginas que se muestran como resultado de las acciones, sobre estas trabajan los diseñadores, con la utilización de las capas (HTML común a todas las páginas). En estas podemos utilizar helpers (son funciones de PHP que devuelven código HTML y que se utilizan en las plantillas, facilitando la creación de las plantillas y producen el mejor código HTML).

Seguridad

Beneficios

- Velocidad de desarrollo.
- Orden del código.
- Estandarización.
- Fácilmente legibilidad para otro programador de Symfony.

Problemas

- Curva de aprendizaje.
- NO se aprende PHP, sino Symfony.
- Las cosas no son mágicas, solo en el caso de que alguien lo programó antes.
- Más capas de abstracción, más tiempo de ejecución de scripts.

Construcción de proyectos/aplicaciones

- Línea de comandos.
- Esquemas de base de datos.
- Modelos de datos.

2.3 Validación de formularios Symfony

Las aplicaciones de la denominada Web 2.0 incluyen numerosas interacciones en el lado del cliente, efectos visuales complejos y comunicaciones asíncronas con los servidores. Todo lo anterior se realiza con JavaScript, pero programarlo manualmente es una tarea tediosa y que requiere de mucho tiempo para corregir los posibles errores. Afortunadamente, Symfony incluye una serie de helpers que automatizan muchos de los usos comunes de JavaScript en las plantillas. La mayoría de comportamientos en el lado del cliente se pueden programar sin necesidad de escribir ni una sola línea de JavaScript. Los programadores solo tienen que ocuparse del efecto que quieren incluir y Symfony se encarga de lidiar con la sintaxis necesaria y con las posibles incompatibilidades entre navegadores.

A continuación se describen las herramientas proporcionadas por Symfony para facilitar la programación en el lado del cliente:

- Los helpers básicos de JavaScript producen etiquetas <script> válidas según los estándares XHTML, para actualizar elementos DOM (Document Object Model) o para ejecutar un script mediante un enlace.
- Prototype es una librería de JavaScript completamente integrada en Symfony y que simplifica el desarrollo de scripts mediante la definición de nuevas funciones y métodos de JavaScript.
- Los helpers de AJAX permiten al usuario actualizar partes de la página web pinchando sobre un enlace, enviando un formulario o modificando un elemento de formulario.

- Todos estos helpers disponen de múltiples opciones que proporcionan una mayor flexibilidad, sobre todo mediante el uso de las funciones de tipo callback.
- Script.aculo.us es otra librería de JavaScript que también está integrada en Symfony y que añade efectos visuales dinámicos que permiten mejorar la interfaz y la experiencia de usuario.
- JSON (JavaScript Object Notation) es un estándar utilizado para que un script de cliente se comunique con un servidor.
- Las aplicaciones Symfony también permiten definir interacciones complejas en el lado del cliente, combinando todos los elementos anteriores. Mediante una sola línea de código PHP (la llamada al helper de Symfony) es posible incluir las opciones de autocompletado, arrastrar y soltar, listas ordenables dinámicamente y texto editable.

Symfony incluye un mecanismo específico de validación de formularios realizado mediante archivos YAML, en vez de utilizar código PHP en la acción.

El funcionamiento básico de la validación en un formulario es que si el usuario introduce datos no válidos y envía el formulario, la próxima página que se muestra debería contener los mensajes de error.

Un validador es una clase que proporciona un método llamado execute(). El método requiere de un parámetro que es el valor del campo de formulario y devuelve verdadero si el valor es válido y falso en otro caso.

Por ejemplo el validador sfStringValidator. Comprueba que el valor introducido es una cadena de texto y que su longitud se encuentra entre 2 límites indicados (definidos cuando se llama al método initialize()).

Las reglas de validación definidas anteriormente se pueden traducir en validadores:

- nombre: sfStringValidator (min=2, max=100)
- email: sfStringValidator (min=2, max=100) y sfEmailValidator
- edad: sfNumberValidator (min=0, max=120)

Archivo de validación

En el archivo de validación, la clave fields define la lista de campos que tienen que ser validados, si son requeridos o no y los validadores que deben utilizarse para comprobar su validez. Los parámetros de cada validador son los mismos que se utilizan para inicializar manualmente los validadores. Se pueden utilizar tantos validadores como sean necesarios sobre un mismo campo de formulario.

Los archivos de validación se encuentran en el directorio validate del módulo de Symfony.

2.4 Validadores estándares de Symfony

Symfony contiene varios validadores ya definidos y que se pueden utilizar directamente en los formularios:

- sfStringValidator
- sfNumberValidator
- sfEmailValidator
- sfUrlValidator
- sfRegexValidator
- sfCompareValidator
- sfPropelUniqueValidator
- sfFileValidator
- sfCallbackValidator

Cada validador dispone de una serie de parámetros y de mensajes de error. Se pueden redefinir fácilmente mediante el método initialize() del validador o mediante el archivo YAML. En las siguientes secciones se describen los validadores y ejemplos de su uso. **(23)**

2.4.1 Validador de cadenas de textos

sfStringValidator. Permite establecer una serie de restricciones relacionadas con las cadenas de texto.

Ejemplo:

sfStringValidator:

values: [valor1, valor2]

values_error: Los únicos valores aceptados son valor1 y valor2.

insensitive: false # *Si vale true, la comparación con los valores no tiene en cuenta mayúsculas y minúsculas.*

min: 2

min_error: Por favor, introduce por lo menos 2 caracteres.

max: 100

max_error: Por favor, introduce menos de 100 caracteres.

2.4.2 Validador de números

sfNumberValidator. Verifica si un parámetro es un número y permite establecer una serie de restricciones sobre su valor.

Ejemplo;

sfNumberValidator:

nan_error: Por favor, introduce un número entero.

min: 0

min_error: El valor debe ser superior a 0.

max: 100

max_error: El valor debe ser inferior a 100.

2.4.3 Validador de email

sfEmailValidator. Verifica si el valor de un parámetro es una dirección de correo válida.

Ejemplo:

sfEmailValidator:

strict: true

email_error: Esta dirección de email no es válida.

La recomendación RFC822 define el formato de las direcciones de correo electrónico. No obstante, el formato válido es mucho más permisivo que el de las direcciones de correo habituales. Según la recomendación, un correo como yo@localhost es una dirección válida, aunque es una dirección que seguramente será poco útil. Si se establece la opción strict: true (que es su valor por defecto) solo se le considera válidas las direcciones de correo electrónico con el formato nombre@dominio.extension. Si la opción strict: false, utiliza las normas de la recomendación RFC822.

2.4.4 Validador de URL

sfUrlValidator. Comprueba si el valor de un campo es una URL válida.

Ejemplo:

sfUrlValidator:

url_error: La URL no es válida.

2.4.5 Validador de expresiones regulares

sfRegexValidator. Permite comparar el valor de un campo con una expresión regular compatible con Perl.

Ejemplo:

sfRegexValidator:

match: No

match_error: Los comentarios con más de una URL se consideran spam.

pattern: /http.*http/si

El parámetro match determina si el parámetro debe cumplir el patrón establecido (cuando vale Yes) o no debe cumplirlo para considerarse válido (cuando vale No).

2.4.6 Validador para comparaciones

sfCompareValidator. Compara dos parámetros de petición. Su mayor utilidad es para comparar dos contraseñas.

Ejemplo:

fields:

password1:

required:

msg: Por favor, introduce una contraseña.

password2:

required:

msg: Por favor, vuelve a introducir la contraseña.

sfCompareValidator:

check: password1

compare_error: Las 2 contraseñas son diferentes.

El parámetro check contiene el nombre del campo cuyo valor debe coincidir con el valor del campo actual para considerarse válido.

Por defecto el validador comprueba que los dos parámetros sean iguales. Se puede utilizar otra comparación indicándola en el parámetro operator. Los operadores de los cuales dispone son >, >=, <, <=, == y !=.

2.4.7 Validador Propel para valores únicos

sfPropelUniqueValidator. Comprueba que el valor de un parámetro de la petición no existe en la base de datos. Validador realmente útil para las columnas que deben ser índices únicos.

Ejemplo:

fields:

nombre:

sfPropelUniqueValidator:

class: Usuario

column: login

unique_error: Ese login ya existe. Por favor, seleccione otro login.

En este ejemplo, el validador busca en la base de datos los registros correspondientes a la clase Usuario y comprueba si alguna fila tiene en su columna login el mismo valor que el parámetro que se pasa al validador.

2.4.8 Validador de archivos

sfFileValidator. Permite restringir el tipo (mediante un arreglo de mime-types) y el tamaño de los archivos subidos por el usuario.

Ejemplo:

fields:

image:

required:

msg: Por favor, sube un archivo de imagen.

file: True

sfFileValidator:

mime_types:

- 'image/jpeg'
- 'image/png'
- 'image/x-png'
- 'image/pjpeg'

mime_types_error: Solo se permiten los formatos PNG y JPEG.

max_size: 512000

max_size_error: El tamaño máximo es de 512Kb.

El atributo file debe valer True para ese campo y el formulario de la plantilla debe declararse de tipo multipart.

2.4.9 Validador de callback

sfCallbackValidator. Delega la validación en un método o función externa. El método que se invoca debe devolver true o false como resultado de la validación.

fields:

numero_cuenta:

sfCallbackValidator:

callback: is_integer

invalid_error: Por favor, introduce un número.

numero_tarjeta_credito:

sfCallbackValidator:

callback: [misUtilidades, validarTarjetaCredito]

invalid_error: Por favor, introduce un número correcto de tarjeta de crédito.

El método o función que se llama recibe como primer argumento el valor que se debe comprobar. Se trata de un método muy útil cuando se quieren reutilizar los métodos o funciones existentes en vez de tener que volver a crear un código similar para la validación.

2.4.10 Validadores Propios

También es posible crear validadores propios. Incluir formularios en las plantillas es muy sencillo gracias a los helpers de formularios que incluye Symfony y a sus opciones avanzadas. Si se definen formularios para modificar las propiedades de un objeto, los helpers de formularios para objetos simplifican enormemente su desarrollo. Los archivos de validación, los helpers de validación y la opción de volver a mostrar los datos en un formulario, les permite reducir el esfuerzo necesario para crear un control estricto de los formularios que sea robusto y a la vez fácil de utilizar por parte de los usuarios. Además, cualquier validación por muy compleja que sea se puede realizar escribiendo un validador propio.

2.5 Estructura y Composición de ExtJS

ExtJS es un framework JavaScript para interfaces avanzadas. Con el tiempo y la experiencia, se ha mejorado y desarrollado, integrándole un sin numero de librerías para infinidad de problemas al programar en la web. Entre las principales cosas que se hacen hay varios: mensajes de advertencia, listas desplegables, validación de campos en formularios, etc. Luego, con el uso, se hace más fácil la programación.

Este panorama se centra en los principales campos en los que la arquitectura ha pasado, o de la existencia de algunas zonas totalmente nuevas de funcionalidad. Cada tema se explica en detalle en las siguientes secciones.

2.5.1 Modelo de Componente

Las capacidades de Component ampliadas y mejoradas, por lo que es una de las clases fundamentales en la arquitectura global. Component proporciona un modelo unificado para la creación de componentes, haciendo uso a la manipulación, la gestión del estado y la destrucción, y en cada uno de los componentes que requiere ExtJS, con estas características principales de Component en 2.0:

Constructor explícito

Component proporciona un constructor base que se no se aplica a ninguna configuración pasada a cualquier subclase, y la función initComponent se puede utilizar todo el camino a través de la cadena de herencia para proporcionar al constructor en cada paso en la jerarquía.

Manejo renderizado

Cada uno de los componentes automáticamente apoya la prestación, y el manejo renderizado, administrado de forma automática. Aun así, todavía tiene la máxima flexibilidad para personalizar el proceso a través de la prestación de eventos.

Manejo de destrucción

Cada componente contiene también una función para destruir. ExtJS gestiona la limpieza automática de basura para recoger y destruir los componentes cuando ya no se necesita.

Estado automático

Incorpora la funcionalidad para el establecimiento y la recuperación del estado de muchos de los componentes.

Interfaz consistente para componente básico

El fundamental comportamiento que puede aplicarse a cualquier componente, como esconder, mostrar, habilitar y deshabilitar se ofrecen por medio de componentes, y estos pueden ser anulados o por cualquier subclase personalizada según las necesidades.

Disponibilidad vía ComponentMgr

Cada componente de Ext registrado en el ComponentMgr para su creación, de modo que puede recuperar cualquier componente en cualquier momento, simplemente llamando Ext.getCmp ('id').

Soporte de Plugins

Cualquier componente puede extenderse mediante el uso de plugins. Un plugins es cualquier clase de inicio con un método que acepte un solo parámetro de tipo Ext.Component. Los complementos se pueden agregar a cualquier componente a través de la opción de configuración de los plugins. Cada plugins puede llamar métodos o responder a los eventos en el Component según sea necesario para ofrecer su funcionalidad.

2.5.1.1 Ciclo de vida de Componentes

En general, la arquitectura de componentes en 2.0 ha sido diseñada para manejar la mayoría de las gestiones de componentes. Con una comprensión profunda del ciclo de vida de los componentes será muy útil. Las siguientes son las más importantes etapas en el ciclo de vida de cada clase componente sobre la base de:

- Inicialización
- Renderización
- Destrucción

2.5.1.2 Componentes XTypes

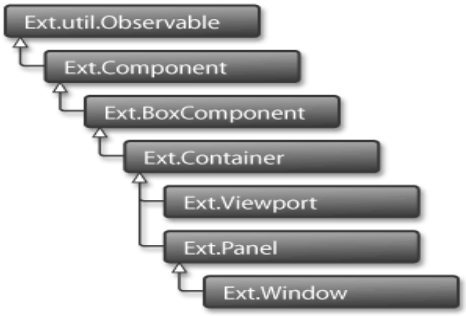
La disposición xtypes se resumen en la cabecera de la clase Componente API. XTypes puede ser usado de manera similar a objeto JavaScript o comparar componentes por tipo utilizando métodos como isXType y getXType. También obtiene un listado xtype de la jerarquía de cualquier componente usando getXTypes.

El poder real de XTypes es la forma en que pueden ser utilizados para la creación de componentes y optimizar la prestación. Cualquier componente se puede crear implícitamente como un objeto de configuración con un xtype especificado, lo que le permite ser declarado y aprobado en el manejo

renderizado sin llegar a ser instanciada como un objeto. No sólo es la prestación diferida, pero la creación del objeto en sí es también para el ahorro de recursos de memoria hasta que sean realmente necesarios. En el complejo, que contiene muchos diseños de componentes anidados, este puede hacer una notable mejora en el rendimiento.

2.5.1.3 BoxComponent

Clase base fundamental que se extiende de Component y proporciona un uso coherente, transversal al modelo del navegador de la aplicación de cualquier componente visual y puede participar en el diseño. BoxComponent maneja transparentemente el tamaño y posicionamiento, automáticamente se ocupan de cualquier navegador de diferencias específicas en el relleno, márgenes y fronteras para producir un modelo de caja a través de cada navegador. Todas las clases contenedores en 2.0 extienden BoxComponent.



2.5.2 Modelos de Contenedores

Existen varias clases básicas disponibles para la creación de diseños de modelos de contenedores y pueden contener otros componentes.

Container

Proporciona el marco fundamental para la contención y la disposición de los componentes, y en esencial para todo el marco visual de ExtJS. Proporciona el diseño y el renderizado, lógica necesaria para la manipulación y el tamaño de anidación de otros componentes, también el mecanismo básico para constantemente añadir componentes en el contenedor. La clase Container nunca debe ser

utilizada directamente, pero se presenta como la clase base para todos los componentes visuales de los contenedores.

Panel

Es el principal contenedor y se usa en el 90% de las tareas. Proporciona los elementos básicos necesarios en la ventana de una aplicación de interfaz de usuario, incluyendo barras de arriba y de abajo para añadir barras de herramientas, menús, cabecera, pie y cuerpo de página. Los paneles pueden caer fácilmente en cualquier contenedor gestionado completamente por ExtJS.

Las siguientes subclases de Panel son widgets primarios en Ext 2.0: GridPanel, TabPanel, TreePanel, FormPanel

Window

Grupo especializado que puede flotar, reducir al mínimo / máximo, restaurar, arrastrar. Se utiliza como la base para la clase de ventana de escritorio como aplicación UIs.

Viewport

La vista en el contenedor es una utilidad que automáticamente hace la clase del cuerpo del documento y tamaños en las dimensiones de la vista en el navegador. Es útil para la creación de aplicaciones de pantalla completa como cambiar el tamaño y el diseño del navegador, cálculos que se gestionan de forma automática.

2.5.3 Layout

Diseño de la arquitectura construida en el nuevo envase y diseño de las clases creadas. Una de las áreas más importantes de mejora en 2.0 es la facilidad y flexibilidad con que pueden crear sofisticados diseños de aplicación. Ext 2.0 ha emitido un completamente renovado sistema de gestión. Se puede mezclar y combinar diferentes contenedores, cada uno con un diseño de anidación diferente, a cualquier nivel que se desea y configuración de todas las opciones que sean específicas para el diseño.

Cuando se crea diseños anidados, con paneles que contienen otros grupos, cada Panel en el diseño debe tener un diseño especificado. Cada diseño de la clase apoya sus propias opciones de configuración.

2.5.3.1 Layout Manager

ContainerLayout: Esta es la clase base para todos los demás elementos de diseño, y el diseño predeterminado para contenedores cuando uno en específico no está definido. ContainerLayout no tiene representación visual, simplemente administra temas, lo que hace cuando es necesario para la manipulación de tareas básicas como el tamaño de memoria intermedia. ContainerLayout en general no debe ser creado directamente, aunque puede ser ampliado para crear diseños personalizados.

AbsoluteLayout: Diseño muy simple que le permite, precisamente, los temas que figuran posición a través de las coordenadas X / Y en relación con el contenedor.

AccordionLayout: Contiene un conjunto de paneles apilados verticalmente-que se puede ampliar y mostrar el contenido. Sólo un grupo puede estar abierto a la vez.

AnchorLayout: Esta disposición es para los elementos de anclaje con relación a los lados. Los elementos pueden ser anclados por porcentaje o por compensaciones de los bordes, y también apoya un lienzo de diseño virtual que puede tener diferentes dimensiones de la física de contenedores.

BorderLayout: Diseño de las regiones con una función de apoyo para la anidación, paneles deslizantes abiertos y cerrados y divisores de la separación de las regiones. Este es el estilo más común para el diseño de la interfaz de usuario principal en aplicaciones de negocios típicas.

CardLayout: Diseño específico utilizado en el caso de que un contenedor pueda contener varios elementos, pero sólo un elemento puede ser visible en un momento dado. Por lo general, este diseño se utiliza para el uso que requieren múltiples y excluyentes páginas de información.

ColumnLayout: Este es el diseño de estilo de elección para la creación de diseños estructurales en un multi-formato de columna, donde el ancho de cada columna puede especificarse como un porcentaje o en pixeles de ancho, la altura, pero se le permite variar en función del contenido.

FitLayout: Estilo simple de diseño que se ajuste a un único tema, a la figura exacta de las dimensiones del contenedor. Normalmente este es el mejor diseño predeterminado para utilizar dentro de los contenedores.

FormLayout: El diseño es una utilidad especialmente diseñada para la creación de formularios de entrada de datos. También se puede utilizar un FormPanel como formulario de presentación de manipulación automática. FormPanel debe utilizar la disposición: «form» (esto no se puede cambiar), por lo que necesitan otras formas de diseño para utilizar estilos anidados.

TableLayout: Genera diseño de estilo estándar HTML de marcas, con el apoyo de la fila y columna que abarca.

2.5.4 Grid

El componente Grid es uno de los contenedores centrales de ExtJS, y ha seguido su evolución como una interfaz de usuario que es ejecutada por la clase GridView, y en 2.0 el GridView ha mejorado mucho. Las principales nuevas características incluyen:

Mejor actuación: La estructura y el código de la prestación del GridView está completamente rediseñado en el rendimiento como la primera prioridad.

- Mejor mirada y percepción: El Grid remodelado se hace visualmente más atractivo que antes.
- La agrupación de las filas: Pueden ser agrupadas las filas en una determinada columna, y reagrupadas por el usuario dinámicamente.
- Grupo resumen de filas: Cada grupo de filas también pueden tener una fila opcional resumen para resumir los datos del grupo.
- Soporta plugins avanzados: El GridView en particular, utiliza la nueva arquitectura de plugins de gran efecto, para el uso con varios plugins hechos y adaptados.

2.5.5 XTemplate

La clase XTemplate ofrece una variedad de funciones de las etiquetas especiales de apoyo a los operadores extremadamente robustos, así como plantilla de procesamiento utilizando complejas estructuras de datos. Aquí está el alto nivel de las características.

- Configuración automática de llenado y el alcance de matrices de conmutación.
- Acceso a objeto de los padres dentro del alcance de la sub–plantilla.
- Índice de matriz de punto variable.
- Básicos operadores matemáticos apoyo en los valores de datos.
- Auto-rendición de arreglo (que contiene los valores no-objeto).
- Básica lógica condicional.
- Capacidad para ejecutar código arbitrario en línea dentro de la plantilla.
- El apoyo a la plantilla de configuración de las propiedades.
- Plantilla de los métodos a través de la configuración de objeto.

2.5.6 DateView

La capacidad de la Vista se ha adoptado para el siguiente nivel con DataView, que se extiende por BoxComponent fácilmente, además de los diseños, también apoya la nueva clase XTemplate como la más potente plantilla para la transformación.

Ambas plantillas de datos usadas para la prestación de apoyo, están obligados a almacenar datos e incorporado en los modelos de selección y eventos. Sin embargo, DataView es un gran paso adelante, porque aprovecha toda la potencia de la nueva arquitectura. Estos son los cambios más significativos:

- Extiende BoxComponent: DataView, se extiende BoxComponent. Esta proporciona muchas ventajas, incluyendo el ciclo de vida y el diseño de las capacidades de componentes BoxComponent.
- Influencias XTemplate: DataView utiliza la clase XTemplate para su plantilla de transformación. Forma mucho más potente de motor de plantilla y permite al DataView emitir datos complejos en una interfaz de usuario personalizada con facilidad.
- Opciones adicionales de configuración: Proporcionando más flexibilidad, con la nueva configuración.

2.6 Elementos básicos de ExtJS

La librería ExtJS ofrece una amplia mejora en el ámbito de aplicaciones web más amigables, para aprender sobre ella consultar en su sitio oficial extjs.com. A través de los conceptos básicos, enseña a crear una página dinámica en muy poco tiempo. Con experiencia en JavaScript y conocimientos básicos del DOM (Document Object Model). Algunas de las tareas más comunes que se tienen que llevar a cabo con JavaScript y aprender cómo hacerlas con ExtJS.

2.6.1 Ext.onRedy

Ext.onReady es probablemente el primer método que se usará en todas las páginas. Este método es llamado automáticamente una vez que el DOM ha sido cargado del todo, garantizando que cualquier elemento de la página al que haga referencia ya estará disponible cuando se ejecute el script. Si da un error de JavaScript, se siguen las instrucciones de la página para hacerlo funcionar.

```
Ext.onReady(function(){
        alert("Congratulations!  You have Ext configured correctly!");
});
```

2.6.2 Element: el corazón de ExtJS

Usando JavaScript tradicional, la selección de un nodo del DOM por el identificador (id) se hace de esta forma:

```
var myDiv = document.getElementById('myDiv');
```

Esto funciona, pero el objeto que es devuelto (un nodo DOM) no ofrece mucho en el sentido de utilidad o conveniencia. Aquí entra el objeto Ext.Element.

Element es el corazón de Ext, ya que gestiona el acceso a los elementos y realiza acciones sobre ellos. El código correspondiente para obtener elemento con Ext por su id es como este (la página HTML de inicio contiene un div con el id "myDiv", en el archivo js escribir esto):

```
Ext.onReady(function(){
        var myDiv = Ext.get('myDiv' ;);
});
```

Obteniendo un objeto Element que contiene muchos de los métodos y propiedades de DOM necesarias, proporcionando una interfaz conveniente, unificada y multinavegador (con acceso directo al nodo DOM a través de Element.dom)

Las acciones más comunes realizadas sobre nodos DOM están integradas en métodos directos independientes del navegador (añadir/borrar clases CSS, añadir/borrar disparadores de eventos, posición, tamaño, animación, arrastrar/soltar, etc.)

Esto significa que se puede hacer todo tipo de cosas útiles con un código mínimo.

```
myDiv.highlight();          // El fondo del elemento se marcará de amarillo y desaparecerá
myDiv.addClass ('red'); // Añade una clase CSS (definida en ExtStart.css)
myDiv.center();             // Centra un elemento en el viewport
myDiv.setOpacity(.25); // Hace al elemento parcialmente transparente.
```

2.6.3 Seleccionando nodos DOM

A veces no es práctico, o posible, acceder a los nodos DOM por su identificador (id). Algunas veces se busca la forma de seleccionar nodos basándose en alguna otra cosa, como un atributo o una clase CSS. Por estas razones, ExtJS proporciona un selector de nodos DOM extremadamente poderoso llamado DomQuery.

DomQuery puede usarse como una librería independiente, pero en el contexto de Ext se usa para seleccionar elementos para que pueda interactuar con ellos a través de la interfaz Element. El objeto Element en si mismo soporta la búsqueda a través del método Element.select, que internamente usa DomQuery para seleccionar los elementos.

Un ejemplo simple de uso está en el archivo HTML, que contiene varios párrafos (etiqueta <p>), ninguno de los cuales tiene identificador (id). Si se quiere seleccionar todos los párrafos y realizar una acción sobre ellos, se puede ejecutar algo como esto:

```
Ext.select('p').highlight ();  //Resalta cada párrafo
```

La selección devuelve un CompositeElement, que proporciona acceso a todos los elementos que contiene a través de la interfaz Element. Esto permite actuar fácilmente sobre todas las instancias devueltas por Element.select sin tener que actuar sobre cada una de forma individual.

DomQuery soporta una amplia variedad de opciones de selección, incluyendo la mayoría de selectores del W3C, CSS3, DOM, XPath básico, atributos HTML entre otros.

2.6.4 Respondiendo a eventos

Para que el código se ejecute en respuesta a eventos o acciones específicas que se quieran gestionar. Se definen gestores de eventos a los que se les puede asignar funciones.

Ejemplo:

```
Ext.onReady(function(){
        Ext.get('myButton').on('click', function(){
                alert("click en el botón" );
        });
});
```

La función obtiene una referencia del elemento con el id 'myButton' y asigna una función para llamarla cada vez que alguien haga click sobre ese elemento.

Element.select permite hacer lo mismo, pero con un grupo entero de elementos de una vez.

Ejemplo:

```
Ext.onReady(function(){
        Ext.select('p').on(' click', function() {
                alert("click en un párrafo");
```

```
    });
});
```

En estos dos ejemplos, la función que gestiona los eventos sin pasar el nombre de ninguna función denominadas “funciones anónimas”. También se puede asignar eventos a funciones con nombre, especialmente útil para reutilizarla en múltiples eventos. Por ejemplo, este código es funcionalmente equivalente al anterior:

```
Ext.onReady(function(){
    var paragraphClicked = function()  {
        alert("click en un párrafo");
    }
     Ext.select('p').on('click', paragraphClicked);
});
```

Son acciones genéricas cuando se lanza un evento, pero para saber que elemento ha sido el que ha lanzado ese evento para poder realizar acciones sobre el mismo, a través del método Element.on, que pasa tres parámetros extremadamente útiles al gestor de eventos.

En los ejemplos anteriores la función del evento ignoraba estos parámetros, pero con un simple cambio puede proporcionar un nuevo nivel de funcionalidad. El primero, y más importante de los parámetros, es qué evento ha ocurrido. Este es actualmente un Ext.event.object. Por ejemplo, el evento del nodo DOM puede ser obtenido con este simple código añadido:

```
Ext.onReady(function(){
    var paragraphClicked = function(e) {
        Ext.get(e.target).highlight();
    }
    Ext.select('p').on('click', paragraphClicked);
});
```

El objetivo es un nodo DOM, así que primero obtiene el elemento correspondiente, luego se realiza cualquier acción sobre el nodo. En este caso, marca visualmente el párrafo.

2.6.5 Usando Widgets

Además del núcleo de la librería ya visto, ExtJS incluye uno de los conjuntos de widgets más ricos de la actualidad. Ejemplo de los más usados.

MessageBox

Para mostrar un texto en una ventana de mensaje. En la llamada al MessageBox, demuestra otro nuevo concepto porque lo que se está pasando en este caso a MessageBox.show() es solo un parámetro: un objeto literal que contiene un conjunto de propiedades y valores.

En JavaScript un objeto literal es un objeto dinámico y genérico que es creado cada vez que se usan los caracteres {y} rodeando una lista de propiedades nombres/valor, y el formato para estas propiedades es [nombre de la propiedad]: [valor de la propiedad].

La principal razón de los literales es la flexibilidad. Nuevas propiedades pueden ser añadidas o eliminadas del objeto literal en cualquier momento, o definidas en cualquier orden, mientras que la firma del método (el número y tipo de parámetros esperados por un método) no cambia nunca. Esto lo hace mucho más conveniente desde la perspectiva del desarrollador final al usar métodos con muchos parámetros opcionales (como en el caso de MessageBox.show()).

Grid

El grid (cuadrícula o rejilla) es uno de los widgets más populares en ExtJS para crear un componente de interfaz extremadamente rico y visualmente complejo con unas pocas líneas de código. Normalmente uno de los primeros para utilizar en proyectos reales, ya que cargan datos de fuentes dinámicas como una base de datos o un servicio web. Para poder cargar un almacén de datos que le dirá a la librería ExtJS subyacente cómo leer y formatear los datos. Lo siguiente es definir el modelo de columnas que simplemente permite configurar las opciones para cada columna del grid. Por último se crea el widget, pasándole el almacén de datos y el modelo de columnas, se renderiza.

Literalmente hay docenas de widgets a escoger en ExtJS, incluyendo composiciones automáticas de páginas, pestañas, menús, barras de herramientas, diálogos, vistas en árbol y mucho más.

2.6.6 Usando AJAX

Al interactuar con una página mediante JavaScript para saber cómo obtener datos de un servidor remoto, normalmente desde una base de datos en el servidor. Hacerlo de forma asincrónica con JavaScript sin recargar la página, es conocido normalmente como AJAX, y ExtJS tiene un soporte perfecto del mismo. Por ejemplo, un acto común es gestionar la interacción del usuario, enviar algo al servidor de forma asincrónica y actualizar el elemento de la interfaz en respuesta a la acción.

El verdadero problema al tratar con el procesamiento de AJAX es la cantidad de código requerido para procesar y formatear correctamente datos realmente estructurados del servidor. Hay varios formatos a

escoger de uso común (normalmente JSON o XML). También hay librerías específicas de un lenguaje disponibles para el proceso de AJAX que pueden funcionar bien con ExtJS, ya que es neutral al lenguaje que se use en el servidor. Siempre que el resultado se envíe a la página en el formato adecuado, ExtJS no se preocupará de lo que pase en el servidor.

2.7 Componentes de ExtJS

ExtJS surgió como YUI-Ext, una extensión de la librería YUI de Yahoo, para recientemente adaptar su código para que funcione igualmente sobre otras librerías como Prototype o JQuery. Se centra en la creación de componentes de interfaz gráfico bastante complejos y con un acabado muy bueno. Pero incluso hasta con muchas menos instrucciones para que estas funcionen, porque siempre se puede fácilmente acceder directamente al DOM de la página y modificar lo que sea necesario. Permiten derivar de sus componentes y modificar comportamientos o bien interceptar los eventos que en general puede ser que el componente no se adapte justo a las necesidades.

2.7.1 Class Ext.Component

Subclasses: BoxComponent, Button, ColorPalette, DatePicker, Editor, BaseItem

Clase base para todos los componentes de Ext. Todas las subclases de Component automáticamente pueden participar en el ciclo de vida del componente de Ext para la creación, utilización y destrucción. También tiene soporte automático de ocultar/mostrar y activar/desactivar el comportamiento. Component deja cualquier subclase incluida en Ext.Container siempre que esta registrada en Ext.ComponentMgr, a fin de que pueda ser accedido en cualquier momento por Ext.getCmp. Todos los dispositivos visuales que se requieren a través del trazado sobre las subclases de Component (o Ext.BoxComponent si es requerido el manejo administrado de modelo de caja).

Cada componente tiene a un xtype específico, el cual es su nombre específico en ExtJS. Esta es la lista de todos los xtypes válidos:

```
xtype            Class-------------------

box              Ext.BoxComponent

button           Ext.Button

colorpalette     Ext.ColorPalette
```

```
component         Ext.Component

container         Ext.Container

cycle             Ext.CycleButton

dataview          Ext.DataView

datepicker        Ext.DatePicker

editor            Ext.Editor

editorgrid        Ext.grid.EditorGridPanel

grid              Ext.grid.GridPanel

paging            Ext.PagingToolbar

panel             Ext.Panel

progress          Ext.ProgressBar

splitbutton       Ext.SplitButton

tabpanel          Ext.TabPanel

treepanel         Ext.tree.TreePanel

viewport          Ext.ViewPort

window            Ext.Window


Toolbar components-------------------------------------

toolbar           Ext.Toolbar

tbbutton          Ext.Toolbar.Button

tbfill            Ext.Toolbar.Fill

tbitem            Ext.Toolbar.Item

tbseparator       Ext.Toolbar.Separator

tbspacer          Ext.Toolbar.Spacer

tbsplit           Ext.Toolbar.SplitButton

tbtext            Ext.Toolbar.TextItem


Form components----------------------------------------
```

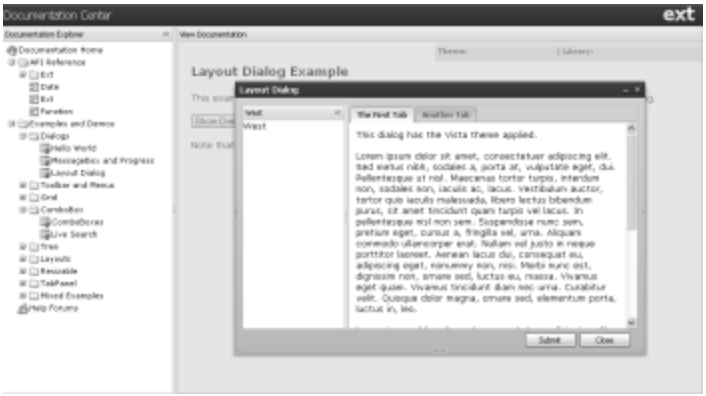
```
form            Ext.FormPanel
checkbox        Ext.form.Checkbox
combo           Ext.form.ComboBox
datefield       Ext.form.DateField
field           Ext.form.Field
fieldset        Ext.form.FieldSet
hidden          Ext.form.Hidden
htmleditor      Ext.form.HtmlEditor
numberfield     Ext.form.NumberField
radio           Ext.form.Radio
textarea        Ext.form.TextArea
textfield       Ext.form.TextField
timefield       Ext.form.TimeField
trigger         Ext.form.TriggerField
```

2.7.2 Class Ext.BoxComponent

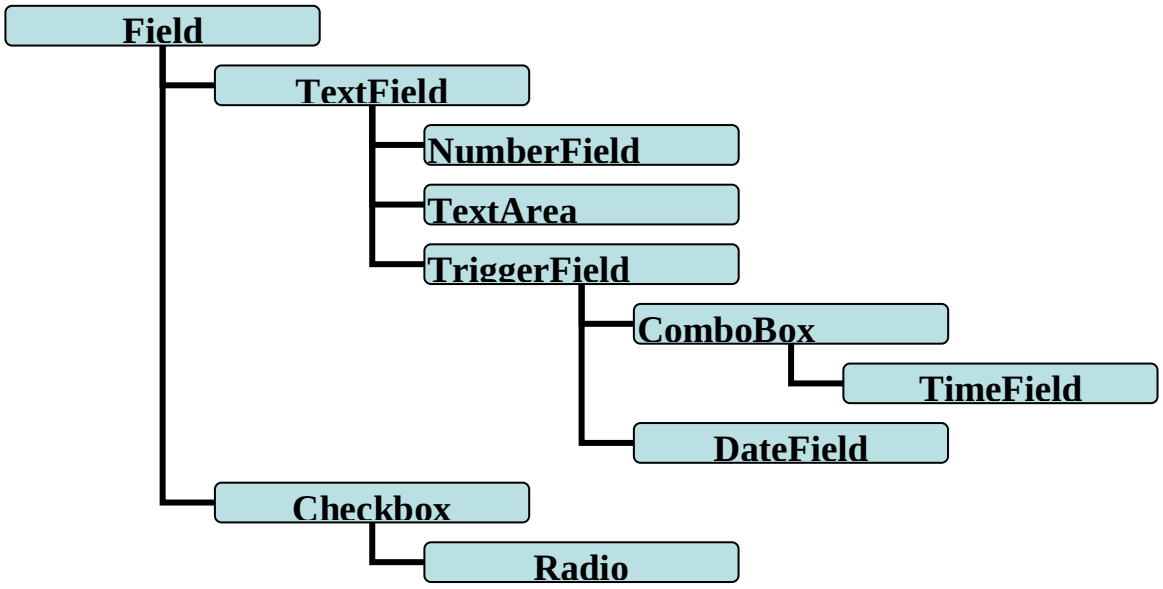
Subclasses: Container, DataView, ProgressBar, Toolbar, Field

Clase base para cualquier Ext.Component visual que usa una caja. BoxComponent provee ajustes automáticos de modelo de caja para dimensionar y posicionar y así trabajar correctamente con el componente volviendo al modelo. Todas las clases del contenido se deben a la subclase BoxComponent para trabajar consistentemente cuándo se anida dentro de otros contenedores de capas.

Esta formidable librería, utiliza uno de los estilos que más se acercan a las aplicaciones de escritorio, mostrados en la figura.



ExtJS es una librería que proporciona una interfaz a las famosas librerías de Yahoo!, JQuery y Prototype + Scriptaculous, su potencia radica en la potente colección de componentes para el diseño de GUI's del lado del cliente haciendo uso extensivo de AJAX. Entre los componentes que esta librería ofrece encontramos cuadros de diálogo, menús, tablas editables, layout, paneles, pestañas y todo lo necesario para construir atractivos desarrollos al estilo de Web 2.0. Cumpliendo con la jerarquía de componentes sobre las que se trabajan las validaciones que son necesarias en los siguientes componentes.



2.7.3 Class Ext.form.Field

Subclasses: Checkbox, Hidden , HtmlEditor , TextField

Clase base para campos del formulario que provee eventos predeterminados, manipuladores y otras funcionalidades.

Opciones de Configuración

disabled : Boolean

Verdadero para desactivar el campo (por defecto false).

focusClass : String

Clase CSS a usar cuando el campo recibe el foco (por defecto "x-form-focus").

inputType : String

El tipo de atributo para el campo, por ejemplo tipo texto, radio, contraseña (por defecto el tipo "text").

invalidClass : String

La clase CSS a usar al señalar un campo inválido (por defecto "x-form-invalid").

invalidText : String

Texto de error a usar al señalar un campo como inválido y ningún mensaje es provisto (por defecto para "el valor en este campo es inválido").

msgFx : String

El efecto cuando es mostrado un mensaje de validación bajo el campo (por defecto 'normal').

msgTarget : String

Posición donde el texto de error debe ser mostrado. Debería ser uno de los siguientes valores (por defecto 'qtip'):

Valor	Descripción
qtip	Muestra un mensaje de aparición automática cuando el usuario esta sobre el dominio.
title	Muestra un mensaje de aparición automática predeterminada de atributo del título del navegador.
under	Adiciona un div al bloque debajo del campo conteniendo el texto de error.
side	Adiciona un icono de error a la derecha del campo con un mensaje de aparición automática.
[element id]	Adiciona el texto de error directamente para el innerHTML del elemento especificado.

name: String

El nombre del atributo del campo HTML.

readOnly : Boolean

Verdadero para marcar el campo como solo lectura en HTML (por defecto false).

validateOnBlur : Boolean

El campo debe validarse cuando pierde foco (por defecto true).

validationDelay : Number

Longitud de tiempo en milisegundos después de que el usuario introduce el dato, empieza hasta que la validación es iniciada (por defecto 250).

validationEvent : String/Boolean

El evento que debería iniciar validación del campo. Si es falso para desactivar validación automática (por defecto "keyup").

value: Mixed

Un valor para darle al campo.

Métodos Públicos

Field(Object config)

Crea un nuevo campo.

Parámetros:

- config : Object. Opciones de configuración.

clearInvalid() : void

Limpia cualquier styles/messages no válidos para este campo.

getName() : String

Devuelve el atributo nombre del campo si esta disponible.

getRawValue() : Mixed

Devuelve el valor original de datos que pueden o no ser un valor válido definido. Para devolver un valor normalizado con getValue().

getValue() : Mixed

Devuelve el valor de datos normalizado (si es indefinido o el campo es vació retorna " "). Para devolver el valor original con getRawValue().

isDirty() : void

Retorna verdadero si el campo se ha cambiado desde que fue originalmente cargado y no está deshabilitado.

isValid(Boolean preventMark) : Boolean

Retorna ya sea o no el valor del campo si es actualmente válido.

Parámetros:

- reventMark : Boolean. Verdadero para desactivar la señal del campo inválido.

Retorna:

- Boolean. Verdadero si el valor es válido, si no falso.

markInvalid(String msg) : void

Marcar este campo como inválido. Muestra el mensaje de validación.

reset() : void

Borra el valor actual del campo, carga en caso de tener un valor por defecto y elimina cualquier mensaje de validación.

setValue(Mixed value) : void

Establece un valor en el campo y lo valida.

validate() : Boolean

Valida el valor del campo. Verdadero si el valor es válido, si no falso.

Eventos Públicos

blur : (Ext.form.Field this)

Usado para cuando el campo pierde foco.

- this : Ext.form.Field

change : (Ext.form.Field this, Mixed newValue, Mixed oldValue)

Usado cuando el campo pierde el foco y si el valor del campo ha cambiado.

- this : Ext.form.Field
- newValue : Mixed. El nuevo valor.
- oldValue : Mixed. El valor original.

focus : (Ext.form.Field this)

Usado cuando el campo recibe foco.

- this : Ext.form.Field

invalid : (Ext.form.Field this, String msg)

Los valores después que el campo es marcado como inválido.

- this : Ext.form.Field
- msg : String. El mensaje de validación.

specialkey : (Ext.form.Field this, Ext.EventObject e)

Usado cuando cualquier tecla guarda relación con la navegación (tab, enter, esc, etc.) o la tecla que se aprieta. Esta función puede apoyarse de Ext.EventObject.getKey para decidir cual tecla se apretó.

- this : Ext.form.Field
- e: Ext.EventObject. El evento del objeto.

valid : (Ext.form.Field this)

Los valores después que el campo ha sido validado sin errores.

- this : Ext.form.Field

2.7.4 Class Ext.form.TextField

Subclasses: Checkbox, Hidden, HtmlEditor, TextField

Campo de texto básico que puede ser usado como un reemplazo directo para los campos de textos tradicionales, o como la clase base para el aporte más sofisticado de control de entrada (Ext.form.TextArea y Ext.form.ComboBox).

Opciones de Configuración

blankText : String

Texto de error a mostrar si la validación fracasa (por defecto "este campo es requerido").

disableKeyFilter : Boolean

Verdadero para desactivar el filtrado de teclas (por defecto false).

emptyClass: String

Clase CSS para aplicar a un campo vacío para darle estilo a emptyText (por defecto 'x-form-empty-field'). Esta clase es automáticamente agregada y eliminada dependiendo del valor actual del campo.

emptyText: String

Texto predeterminado a mostrar en un campo vacío (por defecto null).

maskRe : RegExp

Una expresión regular del cliente aplicada como una máscara que se usará para filtrar teclas que no haga combinación con la máscara (por defecto null).

maxLength: Number

La longitud máxima del campo permitida (por defecto Number.MAX_VALUE).

maxLengthText: String

Texto de error a mostrar si la máxima longitud de validación falla (por defecto "la máxima longitud para este campo es {maxLength}").

minLength : Number

La longitud mínima del campo requerida (por defecto 0).

minLengthText: String

Texto de error a mostrar si la mínima longitud de validación falla (por defecto "la longitud mínima para este campo es {minLength}").

regex: RegExp

Un objeto JavaScript RegExp a ser probado en contra del valor del campo durante la validación (por defecto null). Si está disponible, esta expresión regular (regex) será evaluada sólo después de que todos los validadores regresan verdaderos, y estará aprobado el valor actual del campo. Si la prueba fracasa, el campo será inválido señalado usando regexText.

regexText: String

Texto de error a mostrar si la propiedad regex es usada y la prueba fracasa durante la validación (por defecto " ").

selectOnFocus: Boolean

Verdadero para automáticamente seleccionar cualquier texto existente del campo cuando el campo recibe foco (por defecto false).

validator: Function

Función de validación a ser designada durante la validación del campo (por defecto null). Si está disponible, esta función será designada sólo después de que todos los validadores devuelven verdaderos, y estará aprobado el valor actual del campo y esperado para devolver verdadero si el valor es válido o un mensaje de error si el valor es inválido.

vtype: String

Nombre de tipo de validación definido en Ext.form.VTypes (por defecto null).

vtypeText: String

Mensaje de error a mostrar en lugar del mensaje predeterminado previsto pues el nombre de vtype actualmente se pone para este campo (por defecto " "). Sólo se aplica si el vtype está colocado, si no ignorado.

Métodos Públicos

TextField(Object config)

Crea a un nuevo TextField. Las opciones de configuración.

Parámetros:

- config: Object. Opciones de configuración.

reset() : void

Pone a cero el valor actual del campo para el valor originalmente cargado y cualquier mensaje de validación. También adiciona emptyText y emptyClass si el valor original estuviese en blanco.

selectText([Number start], [Number end]) : void

Selecciona el texto en este campo.

Parámetros:

- start: Number. (Opcional) El índice donde la selección debe empezar (por defecto 0).
- end: Number. (Opcional) El índice donde la selección debe finalizar (por defecto la longitud del texto).

validateValue(Mixed value) : Boolean

Valida un valor según las reglas de validación y pone la marca del campo como inválido si la validación falla.

Parámetros:

- value: Mixed. El valor a validar.

Retorna:

- Bolean. Verdadero si el valor es válido, si no falso.

2.7.5 Class Ext.form.NumberField

Campo de texto numérico que provee el filtrando automático y la validación numérica.

Opciones de Configuración

allowDecimals : Boolean

Falso para prohibir valores decimales (por defecto true).

allowNegative : Boolean

Falso para impedir entrar en el campo un signo de menos (por defecto true).

baseChars : String

La base de caracteres colocada para evaluar como números válidos (por defecto ' 0123456789 ').

decimalPrecision : Number

La máxima precisión a mostrar después del separador decimal (por defecto 2).

decimalSeparator : String

Carácter(s) para establecer como el decimal separador (por defecto '**.** ').

fieldClass : String

La clase CSS predeterminada para el campo (por defecto "x-form-field x-form-num-field").

maxText : String

Texto de error a mostrar si la máxima validación del valor falla (por defecto "el valor máximo para este campo es {maxValue}").

maxValue : Number

El máximo valor permitido (por defecto Number.MAX_VALUE).

minText : String

El texto de error a mostrar si la validación mínima del valor falla (por defecto "el valor mínimo para este campo es {minValue}").

minValue : Number

El mínimo valor permitido (por defecto Number.NEGATIVE_INFINITY).

nanText : String

Texto de error a mostrar si el valor no es un número válido. Por ejemplo, esto puede ocurrir si un carácter válido como '**.**' o '**-**' se queda en el campo sin número (por defecto "{value} no es un número válido").

Métodos Públicos

NumberField(Object config)

Crea a un nuevo campo numérico.

Parámetros:

- config: Object. Opciones de configuración.

2.7.6 Class Ext.form.TextArea

Área de Texto que puede ser usada como un reemplazo directo para los campos tradicionales de texto, y soporta características para autodimensionar el área de texto.

Opciones de Configuración

autoCreate : String/Object

Una parte del elemento DomHelper, o verdadero para una parte predeterminada del elemento (por defecto {tag: "textarea", style: "width: 100px, height: 60px;", autocomplete: "off"}).

preventScrollbars : Boolean

Verdadero para impedir aparecer barras de desplazamiento a pesar de cuánto texto está en el campo (el equivalente para setting overflow: hidden, por defecto false).

Métodos Públicos

TextArea(Object config)

Crea a una nueva área de texto.

Parámetros:

- config: Object. Opciones de configuración.

2.7.7 Class Ext.form.TriggerField

Subclasses: ComboBox, DateField

Provee una envoltura conveniente para los campos de texto que suma un botón capaz de hacer clic. El TriggerField no tiene acción predeterminada, para asignar una función para implementar el manipulador del trigger por onTriggerClick. Se puede crear en un TriggerField directamente, tan exactamente como un ComboBox para el cual puede proveer una implementación

Ejemplo:

var trigger = **new** Ext.form.TriggerField();

trigger.onTriggerClick = myTriggerFn;

trigger.applyToMarkup('my-field');

Sin embargo, en general es más probable utilizar un TriggerField como la clase base para un componente reusable. Ext.form.DateField y Ext.form.ComboBox son ejemplos perfectos de este tipo de componente.

Opciones de Configuración

autoCreate : String/Object

Una parte del elemento DomHelper, o verdadero para una parte predeterminada del elemento (por defecto {tag:”input”, type:"text", size: "16”, autocomplete: "off"}).

Public Methods

TriggerField(Object config)

Crea a un TriggerField nuevo.

Parámetros:

- config: Object

  Las opciones de configuración (válido {Ext.form.TextField} Las opciones de configuración también son aplicadas al TextField base).

onTriggerClick(EventObject e) : void

La función que debe manejar el evento de clic del trigger. Este método no hace nada por defecto hasta pasar sobre la disposición de por una función que implementa.

Parámetros:

- e: EventObject

2.7.8 Class Ext.form.ComboBox

Subclasses: TimeField

Un control del ComboBox es el apoyo para el autocompletamiento, remoto, compaginado y muchos otros rasgos presentes en el TimeField.

Opciones de Configuración

allQuery : String

La consulta de texto para enviar al servidor para devolver todos los archivos para la lista sin filtrarse (por defecto” ”).

editable : Boolean

Falso para impedirle al usuario teclear el texto directamente en el campo, sólo como un tradicional seleccione (por defecto true).

forceSelection : Boolean

Verdadero para restringir el valor seleccionado a uno de los valores en la lista, falso para permitirle al usuario poner el texto arbitrario en el campo (por defecto false).

minChars : Number

Número mínimo de caracteres que el usuario debe teclear antes del autocompletamiento y que este se active (por defecto 4 si es remoto o 0 si es local, no se aplica si el ComboBox tiene la propiedad editable = false)

mode : String

Modo 'local' si el ComboBox carga los datos locales (por defecto 'remoto' cargados del servidor).

Métodos Públicos

ComboBox(Object config)

Crea un nuevo ComboBox.

Parámetros:

- config: Object. Opciones de configuración.

clearValue() : void

Limpia cualquier text/value actualmente puesto en el campo.

getValue() : String

Devuelve el valor del campo actualmente seleccionado o la cadena vacía si no tiene valor o el valor por defecto.

2.7.9 Class Ext.form.TimeField

Campo de tiempo que provee aporte a través del tiempo y validación de forma automática.

Opciones de Configuración

altFormats : String

Los formatos múltiples de fecha separados por "|" para intentar analizarla gramaticalmente con el valor escrito por el usuario y no corresponde al formato definido (por defecto 'm/d/Y | m-d-y | m-d-Y | m/d | m-d | d').

format : String

Formato de fecha predeterminada del que puede ser sobrescrito para soporte de localización. El formato debe ser válido según Date.parseDate (por defecto 'm/d/y').

increment : Number

El número de minutos entre cada valor de tiempo en la lista (por defecto 15).

invalidText : String

Texto de error a mostrar cuando el tiempo en el campo es inválido (por defecto '{value} no es un tiempo válido - debe estar en el formato {format}').

maxText : String

Texto de error a mostrar cuando el tiempo es mayor que el valor de maxValue (por defecto 'El tiempo en este campo debe ser menor o igual que {0}').

maxValue : Date/String

El máximo permitido de tiempo. Puede estar ya sea un objeto de fecha JavaScript o una fecha en un formato válido (por defecto null).

minText : String

Texto de error a mostrar cuando el tiempo es menor que el valor de minValue (por defecto ' El tiempo en este campo deben ser mayor o igual que la fecha {0} ').

minValue : Date/String

Mínimo tiempo permitido. Puede estar ya sea un objeto de fecha JavaScript o una fecha de en un formato válido (por defecto null).

Métodos Públicos

TimeField(Object config)

Cree a un nuevo campo de tiempo.

Parámetros:

- config: Object. Opciones de configuración.

2.7.10 Class Ext.form.DateField

Provee un campo de aporte de fecha de un Ext.DatePickerDropdown y validación automática de fecha.

Opciones de Configuración

altFormats : String

Los formatos múltiples de fecha se separados por "|" ahí para intentar analizarla gramaticalmente con el valor escrito por el usuario y no corresponde al formato definido (por defecto 'm/d/Y | m-d-y | m-d-Y | m/d | m-d | d').

autoCreate : String/Object

Una parte del elemento DomHelper, o verdadero para una parte predeterminada del elemento (por defecto {tag: "input", type: "text", size: "10", autocomplete: "off"})

disabledDates : Array

Lista de fechas a desactivar. Estas cadenas se usarán para construir una expresión normal dinámica que son muy poderosas.

Ejemplos:

- ["03/08/2003", "09/16/2003"] Desactivarían esas fechas exactas.
- ["03/08", "09/16"] Desactivarían esos días para cada año.
- ["^03/08"] Sólo correspondería al comienzo (útil si usted usa en seco los años).
- ["03/../2006"] Desactivaría todos los días en marzo 2006.
- ["^03"] Desactivaría todos los días de marzo.

Para soportar expresiones normales, si está usando un formato de fecha que tiene "**.**" en eso, se tendrá que librarse del punto al restringir las fechas.

Ejemplo: ["03\\.08\\.03"].

disabledDatesText : String

Texto del mensaje de aparición automática a mostrar cuándo la fecha esta sobre una fecha deshabilitada (por defecto 'Disabled').

disabledDays : Array

Lista de días a desactivar, basados en 0. Por ejemplo, 0, 6 desactiva domingo y sábado (por defecto null).

disabledDaysText : String

Mensaje de aparición automática a mostrar cuando la fecha esta sobre un día deshabilitado (por defecto 'Disabled').

format : String

Formato de fecha predeterminado, que puede ser escrito para soporte de localización. El formato debe ser válido según Date.parseDate (por defecto 'm/d/y').

invalidText : String

Texto de error a mostrar cuando es la fecha un campo inválido (por defecto '{value} no es una fecha válida - debe estar en el formato {format}').

maxText : String

Texto de error a mostrar cuando la fecha del campo está por detrás de maxValue (por defecto ' La fecha en este campo debe estar antes de {maxValue} ').

maxValue : Date/String

Valor máximo de fecha permitida. Puede estar ya sea un objeto de fecha JavasScript o una fecha en un formato válido (por defecto null).

minText : String

Texto de error a mostrar cuando la fecha del campo está por delante de minValue (por defecto ' La fecha en este campo debe estar después de {minValue} ').

minValue : Date/String

Valor mínimo de fecha permitida. Puede estar ya sea un objeto de fecha JavaScript o una fecha en un formato válido (por defecto null).

Métodos Públicos

DateField(Object config)

Crea a un nuevo DateField.

Parámetros:

- config: Object. Opciones de configuración

getValue() : Date

Retorna el valor de fecha actual del campo de fecha.

Parámetros:

- None.

Retorna:

- Date. El valor de la fecha.

setValue(String/Date date) : void

Establece el valor del campo de fecha. Se puede pasar un objeto de fecha o cualquier valor que se analice gramaticalmente como una fecha válida, utilizar DateField.format como el formato de fecha, según las mismas reglas como Date.parseDate (el formato predeterminado usado es 'm/d/y').

Ejemplo:

```
//todas estas llamadas establecen el mismo valor de fecha (4 de mayo, 2006)
// Pasar un objeto de fecha
var dt = new Date('5/4/06');
dateField.setValue(dt);
//Pasar una cadena de fecha (formato por defecto):
dateField.setValue('5/4/06');
//Pasar una cadena de fecha (formato específico):
dateField.format = 'Y-m-d';
dateField.setValue('2006-5-4');
```

Parámetros:

- date: String/Date. La fecha o una cadena de fecha válida.

2.8 Class Ext.form.Vtypes

Las definiciones de validación. Las validaciones proporcionadas son básicas y pensadas para ser fácilmente personalizadas y extensibles.

Propiedades Públicas

alphaMask : RegExp

Máscara de filtro de pulsación a ser aplicada en la en la validación de valores alfa.

alphaText : String

Mensaje de error para desplegar cuando la función de validación alfa retorna falso.

alphanumMask : RegExp

Máscara de filtro de pulsación a ser aplicada en la validación de valores alfanuméricos.

alphanumText : String

Mensaje de error para desplegar cuando la función de validación alfanumérica retorna falso.

emailMask : RegExp

Máscara de filtro de pulsación a ser aplicada en la validación del correo electrónico.

emailText : String

Mensaje de error para desplegar cuando la función de validación de correo electrónico retorna falso.

urlText : String

Mensaje de error para desplegar cuando la función de validación de URL retorna falso.

Métodos Públicos

alpha(String value) : void

Función usada para validar valores alfa.

Parámetros:

- value: String. Valores alfa.

alphanum (String value) : void

Función usada para validar valores alfanuméricos.

Parámetros:

- value: String. Valores alfanuméricos.

email (String value) : void

Función usada para validar direcciones de correo electrónico.

Parámetros:

- value: String. La dirección de correo.

url(String value) : void

Función usada para validar URLs

Parámetros:

- value: String. La URL

2.9 Conclusiones

Como conclusión primordial de este capítulo con el estudio de las validaciones que realiza el propio framework Symfony en su capa de presentación, y la estructura, composición de los distintos

componentes de la librería ExtJS para un mejor desempeño de las validaciones que se utilizan dentro del mismo.

CAPÍTULO 3: DESARROLLO DE LAS VALIDACIONES

3.1 Introducción

El mundo se mueve por códigos. Todas nuestras acciones y transacciones por simples que parezcan, poseen miles de combinaciones numéricas que interactúan de forma invisible a nuestros ojos cada vez que realizamos una operación. En fin, el mundo se rinde ante los datos.

Con el propio estudio del capítulo anterior, para la implementación de las validaciones que necesita la Aduana General de la República, en su desarrollo, el empleo de las validaciones contenidas por el framework Symfony y ExtJS, se requieren para la realización responsable de la incorporación de datos al sistema una mayor rigurosidad de los mismos, así como una confirmación y revisión del ingreso de todos los datos tratados por ambas partes. Por esta razón se propone el desarrollo del framework que contenga en su realización la validación de los datos.

3.2 Guía para la integración de ExtJS - Symfony

El trabajando hecho para la integración del ExtJS al framework Symfony con el objetivo de mostrar una forma de integrar estos dos frameworks para así obtener una interfaz única con la cual gestionar datos en una base de datos y mejores interfaces de aplicación.

El resultado muy útil sin la necesidad de recargar la página, ya que toda la comunicación cliente servidor se realiza mediante AJAX y los datos se transmiten en formato JSON.

3.2.1 Integración del ExtJS - Symfony

Copiar de la carpeta ext-2.0.2, para el directorio local del escritorio.

Teniendo el proyecto creado a través de Symfony. En caso de no tenerlo se crea.

- Dentro del proyecto creado, Symfony crea una estructura de directorios que se muestra a continuación:

  apps/
  batch/
  cache/
  config/

data/
doc/
lib/
log/
plugins/
test/
web/

Dentro del directorio del proyecto, dentro de la carpeta web/js.

- El archivo que esta en la carpeta Ext del escritorio, ext-2.0.2/ext-all.js es copiado para la carpeta que se encuentra dentro del directorio web/js.
- El archivo que esta en la carpeta Ext del escritorio, ext-2.0.2/adapter/ext/ext-base.js es copiado para la carpeta que se encuentra dentro del directorio web/js.

Dentro del directorio del proyecto, dentro de la carpeta web/css.

- La carpeta sourse que esta en la carpeta Ext del escritorio, es copiada para la carpeta que se encuentra dentro del directorio web/css.
- Para el cual el archivo ext-all.css que se encuentra en el directorio web/css/resources/css/ext-all.css es agregado.

Teniendo la aplicación creada a través de Symfony. En caso de no tenerlo se crea.

- Dentro de la raíz del proyecto se encuentra el directorio apps/miaplicacion. Por defecto crea una configuración básica de la aplicación y una serie de directorios:

apps/
  miaplicacion/
    config/
    i18n/
    lib/
    modules/
    templates/

Dentro del directorio del proyecto, dentro de la carpeta apps/
miaplicacion/
config/

- El archivo view.yml se modifica en las siguientes líneas de código:

stylesheets: [main]

javascripts: []

Entre los corchetes de la línea de código de javascript se escribe los nombres de los archivos js agregados anteriormente en el directorio web/js, el nombre separando por un espacio después de la coma.

- Quedando así de esta forma la línea de código: javascripts: [ext-all, ext-base]

Entre los corchetes de la línea de código de stylesheets se escribe el nombre de los archivos css agregados anteriormente en el directorio web/css, el nombre separado por un espacio después de la coma.

- Quedando así de esta forma la línea de código: stylesheets: [main, ext-all]

3.2.3 Symfony y ExtJS

Los framewoks se han caracterizado en los últimos tiempos por presentarse en versiones con los requerimientos mínimos para su funcionamiento y cualquier otra funcionalidad que se requiera se añade a través de extensiones o plugins. Symfony no es una excepción a esto, por eso a un usuario común le puede parecer después de una instalación básica de este framework que carece de muchos de los elementos que necesitaría.

La mayoría de estos elementos se encuentran disponibles en extensiones que hacen uso de ficheros agregados. Symfony cuenta con una serie de funciones agrupadas en varios archivos que han ido consolidándose en una especie de librería, pero que brinda funcionalidades muy básicas.

A través del uso de ExtJS como librería de JavaScript se puede ampliar el espectro de las extensiones a usar en Symfony y así mejorar muchas de sus funcionalidades.

3.3 Objetos Nativos de JavaScript

JavaScript define algunos objetos de forma nativa, es decir, que pueden ser utilizados directamente por los scripts sin tener que declararlos. Además JavaScript define muchas otras clases de tipo como por ejemplo Object, Number, Boolean, String, Function, Array, Date y RegExp presentes dentro de ExtJS. **(13)**

3.3.1 Class Array

JavaScript permite definir los arreglos de forma abreviada y también de forma tradicional mediante la clase Array:

Métodos Públicos

indexOf(Object o) : Number

Verifica si un objeto específico se encuentra en el arreglo.

Parámetros:

- o: Object. El objeto a buscar.

Retorna:

- Number. El índice del Objeto en el arreglo(-1 en caso de no encontrarlo)

remove(Object o) : Array

Elimina el objeto especificado en el arreglo. Si el objeto no se encuentra no pasa nada.

Parámetros:

- o: Object. El objeto a eliminar.

Retorna:

- Array. El arreglo.

3.3.2 Class String

Estas funciones están disponibles como los métodos estáticos en el objeto JavaScript String.

Métodos Públicos

espace(String string): String

<static> Cadena de escape aprobada para ' y \

Parámetros:

- string : String. La cadena de escape.

Retorna:

- String. La cadena evadida

format(String string, String value1, String value2) : String

<static> Deja definir una cadena y pasar un número arbitrario de argumentos para reemplazar los tokens. Cada token debe ser único, y debe incrementar en el formato {0}, {1}, etc.

Ejemplo:

var cls = *'my-class'*, text = *'Some text'*;

var s = String.format(*'{1}'*, cls, text); //s ahora contiene la cadena: *'Some text'*

Parámetros:

- string : String. La cadena a ser formateada
- value1 : String. El valor para reemplazar token{0}
- value2 : String

Retorna:

- String. La cadena formateada.

leftPad(String string, Number size, [String char]) : String

<static> Rellena el lado izquierdo de una cadena con un carácter especificado. Esto es especialmente muy apropiado para normalizar cadenas de número y de fecha. Ejemplo:

var s = String.leftPad(*'123'*, 5, *'0'*);

//s ahora contiene la cadena: *'00123'*

Parámetros:

- string : String. La cadena original.
- size : Number. La longitud total de la cadena de salida
- char : String. (opcional) El carácter con el cual rellena la cadena original (por defecto es la cadenas vacía " ")

Retorna:

- String. La cadena completada.

toggle(String value, String other) : String

Cambia una cadena entre dos valores alternantes. El valor aprobado es comparado con la cadena actual, si son iguales, el otro valor en el que se pasó es devuelto. Si son diferentes, el primer valor pasado adentro es devuelto. Este método devuelve el valor nuevo pero no cambia la cadena actual.

// alternar el orden

sort = sort.toggle(*'ASC'*, *'DESC'*);

// instanciando una logica condicional:

sort = (sort == *'ASC'* ? *'DESC': 'ASC'*);

Parámetros:

- value : String. El valor a comparar con la cadena actual.
- other : String. El valor nuevo a usar si la cadena ya iguala el primer valor pasado adentro.

Retorna:

- String. El nuevo valor.

trim() : String

Elimina espacios de cualquier cadena, dejando espacios dentro de la cadena intacta.

Ejemplo:

var s = *' foo bar '*;

alert(*'-'* + s + *'-'*); *//*alerta *"- foo bar -"*

alert(*'-'* + s.trim() + *'-'*); //alerta *"-foo bar-"*

3.3.4 Class Number

Métodos Públicos

constrain(Number min, Number max) : Number

Chequea que el número actual está o no dentro de un rango deseado. Si el número está dentro del rango es devuelto el número, de otra manera el mínimo o máximo valor es devuelto cuando se excede por un rango. Este método devuelve el valor contenido pero no cambia el número actual.

Parámetros:

- min: Number. El mínimo número en el rango.
- max: Number. El máximo número en el rango.

Retorna:

- Number. El valor contenido esta fuera del rango, de otra manera el valor actual.

3.3.5 Class Date

Entre las utilidades que proporciona JavaScript, se encuentra la clase Date que permite representar y manipular valores relacionados con fechas. La fecha analizada gramaticalmente y la sintaxis del formato es un subconjunto de la función de fecha de PHP(), y los formatos que son soportados le proveen los resultados equivalentes a sus versiones PHP. Lo siguiente es una lista de todos los formatos actualmente soportados:

Formato	Descripción	Ejemplo de valores retornados
d	Días del mes, 2 dígitos con ceros.	01 a 31

D	Representación en texto corto del día de la semana.	Mon a Sun
j	Días del mes, sin ceros.	1 a 31
l	Representación completa del día de la semana.	Sunday a Saturday
N	ISO-8601 Representación numérica de los días de la semana.	1 (Monday) hasta 7 (Sunday)
S	Ordinal sufijo para el día del mes, 2 caracteres.	st, nd, rd or th.
w	Representación numérica de los días de la semana.	0 (Sunday) a 6 (Saturday)
z	Los días del año (comenzando de 0)	0 a 364 (365 en años bisiestos)
W	ISO-8601 números de semana del año, semanas comenzando el lunes.	01 a 53
F	Representación textual del mes.	January a December
m	Representación numérica del mes, con 0. .	01 a 12
M	Representación en texto corto del mes.	Jan a Dec
n	Representación numérica del mes, sin 0.	1 a 12
t	El número de días en el mes dado.	28 a 31
L	Si es un año bisiesto.	1 si es año bisiesto, 0 otro caso.
Y	Representación del año en texto completo, 4 dígitos.	Ejemplo: 1999 o 2003
y	Representación de un año con dos dígitos.	Ejemplo: 99 o 03
a	Minúsculas Ante meridiano y Pasado meridiano.	am o pm
A	Mayúsculas Ante meridiano y Pasado meridiano.	AM o PM
g	12-horas, formato de hora sin ceros.	1 a 12
G	24-horas, formato de hora desde ceros.	0 a 23
h	12-horas, formato de hora con ceros.	01 a 12
H	24-horas, formato de hora con ceros.	00 a 23
i	Minutos, con ceros	00 a 59
s	Segundos, con ceros	00 a 59
u	Milisegundos, con ceros	001 a 999

Propiedades Públicas

Date.DAY : String

Date.HOUR : String

Date.MILLI : String

Date.MINUTE : String

Date.MONTH : String

Date.SECOND : String

Date.YEAR : String

<static> Intervalo constante de fecha.

Date.dayNames : Array

<static> Arreglo de nombres textuales de día, por ejemplo.

Date.dayNames = ['SundayInYourLang', 'MondayInYourLang',...];

Date.monthNames : Array

<static> Arreglo de nombres textuales de mes.

Ejemplo:

Date.monthNames = ['JanInYourLang', 'FebInYourLang',...];

Date.monthNumbers : Object

<static> Objeto de números de mes basado en ceros (con los nombres breves de mes como llaves.

Nota: keys: son sensibles a las mayúsculas-minúsculas).

Ejemplo.

Date.monthNumbers = {'ShortJanNameInYourLang':0, 'ShortFebNameInYourLang':1...};

Public Methods

Date.getMonthNumber(String name) : Number

<static> Obtiene el número del mes basado en ceros, dado de forma corta/completa el nombre de mes.

Date.getShortDayName(Number day) : String

<static> Obtiene el nombre breve del día para el número dado.

Date.getShortMonthName(Number month) : String

<static> Obtiene el nombre breve del mes para el número dado.

Date.parseDate(String input, String format) : Date

<static> Analiza gramaticalmente la cadena aprobada usando el formato especificado. Esta función espera las fechas en formato normal del calendario, los meses son basado 1 (1 = enero). Cualquier parte del formato de fecha que no está especificado dejará incumplida para el valor actual de fecha para esa parte. Las partes de tiempo también pueden estar especificadas, pero pueden fallar para 0. La cadena de fecha precisamente debe corresponder a la especificada cadena del formato o lo analiza gramaticalmente y operación fracasa.

Ejemplo:

//dt = Fri May 25 2007 (día actual)

var dt = **new** Date();

*//dt = Thu May 25 2006 (today's month/day **in** 2006)*

dt = Date.parseDate(*"2006"*, *"Y"*);
//dt = Sun Jan 15 2006 (parte especificada del día)
dt = Date.parseDate(*"2006-01-15"*, *"Y-m-d"*);
//dt = Sun Jan 15 2006 15:20:01 GMT-0600 (CST)
dt = Date.parseDate(*"2006-01-15 3:20:01 PM"*, *"Y-m-d h:i:s A"*);
add(String interval, Number value) : Date
Provee un método conveniente de aritmética básica de fecha. Este método no modifica la instancia de Fecha designada - crea y devuelve una instancia nueva de fecha conteniendo el valor resultante.
Ejemplos:
// Uso básico:
var dt = **new** Date(*'10/29/2006'*).add(Date.DAY, 5);
document.write(dt); *//Retorna 'Fri Oct 06 2006 00:00:00'*
// Los valores negativos se sustraerán correctamente:
var dt2 = **new** Date(*'10/1/2006'*).add(Date.DAY, -5);
document.write(dt2); *//Retorna 'Tue Sep 26 2006 00:00:00'*
//Puede concatenar varias llamadas juntas en una línea
var dt3 = **new** Date(*'10/1/2006'*).add(Date.DAY, 5).add(Date.HOUR, 8).add(Date.MINUTE, -30);
document.write(dt3); //Retorna *'Fri Oct 06 2006 07:30:00'*
between(Date start, Date end) : Boolean
Inspecciona si esta fecha se encuentra entre el día de inicio y el día final.
clearTime(Boolean clone) : Date
Limpia cualquier información de tiempo de esta fecha.
clone() : Date
Crea y devuelve una instancia nueva de Fecha con el valor exacto de fecha como la instancia designada. La intención es crear una variable nueva que no modificará la instancia original, creando un clon. El ejemplo para formar correctamente un clon de fecha:
//camino equivocado:
var orig = **new** Date(*'10/1/2006'*);
var copy = orig;
copy.setDate(5);
document.write(orig); *//Retorna 'Thu Oct 05 2006'!*
//camino correcto:

var orig = **new** Date(*'10/1/2006'*);
var copy = orig.clone();
copy.setDate(5);
document.write(orig); //Retorna *'Thu Oct 01 2006'*
format(String format) : String
Formatea una fecha dado el formato de cadena.
getDayOfYear() : Number
Devuelve el número de días del año, ajustado para el año bisiesto. 0 hasta 364 (365).
getDaysInMonth() : Number
Devuelve el número de días en el mes actual, ajustado para el año bisiesto.
getElapsed([Date date]) : Number
Devuelve el número de milisegundos entre esta fecha y otra.
getFirstDateOfMonth() : Date
Devuelve la fecha del primer día del mes.
getFirstDayOfMonth() : Number
Devuelve el primer día del mes actual, ajustado para el año bisiesto. El valor devuelto es el índice numérico de día dentro de la semana (0-6) que puede ser usado en conjunción con monthNames para recuperar el nombre del día.
Ejemplo:
var dt = **new** Date(*'1/10/2007'*);
document.write(Date.dayNames[dt.getFirstDayOfMonth()]); //salida: *'Monday'*
getGMTOffset(Boolean colon) : String
Devuelve el offset de GMT de la fecha actual (equivalente para el formato específico 'O').
La cadena de offset de 4 caracteres prefijada con + o - (ejemplo. '- 0600 ').
getLastDateOfMonth() : Date
Devuelve la fecha del último día del mes.
getLastDayOfMonth() : Number
Devuelve el último día del mes actual, ajustado para el año bisiesto. El valor devuelto es el índice numérico de día dentro de la semana (0-6) que puede ser usado en conjunción con monthNames para recuperar el nombre del día.
Ejemplo:
var dt = **new** Date(*'1/10/2007'*);

document.write(Date.dayNames[dt.getLastDayOfMonth()]); //salida: *'Wednesday'*

getSuffix() : String

Devuelve el sufijo ordinal inglés del día actual (Equivalente para formato específico. 'st, 'nd', 'rd' o 'th')

getTimezone() : String

Devuelve la abreviación de la zona horaria de la fecha actual (el equivalente para el formato específico 'T').

getWeekOfYear() : Number

Devuelve el número de semana ISO-8601 del año. (Equivalente para el formato específico ' W ', pero sin empezar de cero, desde 1 a 53)

isLeapYear() : Boolean

Si la fecha actual está en un año bisiesto.

3.4 Validaciones comunes de la Aduana

En el propio ExtJS se encuentran validaciones dentro de los propios componentes de formularios poseen Opciones de Configuración que se aplican de forma directa en las validaciones de la Aduana.

3.4.1 Validaciones de String

xtype : 'textfield'

3.4.1.1 Validar solo letras

maskRe : new RegExp(/^([a-zA-Z])$/)

3.4.1.2 Validar solo números

maskRe : new RegExp(/^([0-9])$/)

3.4.1.3 Validar si un e-mail es valido.

vtype : 'email',

vtypeText : 'Correo no válido'

3.4.2 Validaciones de números
xtype : 'numberfield'

3.4.2.1 Validar solo números decimales

allowDecimals : true

3.4.2.2 Validar números enteros

baseChars : '1234567890',
nanText : 'No es un número entero valido',
allowDecimals : false

3.4.2.3. Validar rango de un número

minValue : 0,
minText :'Valor Mínimo 0',
maxValue : 100,
maxText :'Valor Máximo 100'

3.4.2.4 Validar porciento

allowDecimals : false,
allowNegative : false,
minValue : 0,
minText : 'Valor Mínimo 0',
maxValue : 100,
maxText : 'Valor Máximo 100'

3.4.2.5 Validar dígitos de Carnet Identidad

minLength : 11,
minLengthText : "Cantidad de números 11",
maxLength : 11,
maxLengthText : "Cantidad de números 11"

3.4.3 Validaciones de fechas

xtype : 'datefield'

3.4.3.1 Validar el formato de la fecha

format: 'd/m/Y',
readOnly: true

3.4.3.2 Validar año introducido este entre (1900-2900)

minValue : 1900,
minText : 'Año mayor a 1900',
maxValue : 2900,
maxText : 'Año menor a 2900'

3.4.3.3 Fecha menor igual al día de hoy

maxValue :new Date

3.4.3.4 Fecha mayor al día de hoy

minValue : new Date

3.4.3.5 Fecha inicio sea mayor igual que la fecha fin

Como se puede tratar la fecha de un DateField que sigue de otro a continuación.
Compara que la fecha fin tenga como valor mínimo el valor marcado como la fecha de inicio.
Ejemplo:

listeners: {
'change': function(dateCombo){
Ext.getCmp('id').minValue = dateCombo.getValue(); // Obtiene el componente con el identificador ‘id’ y le asigna a su configuración como mínimo valor el valor del DateCombo.
} }

3.5 Clases

Los objetos vistos hasta el momento son simplemente una colección de propiedades y métodos que se definen para cada objeto individual. Sin embargo, en la programación orientada a objetos, el concepto fundamental es el de clase. Los lenguajes permiten definir una clase a partir de la cual se crean objetos de ese tipo de clase. JavaScript no permite la creación de clases del mismo tipo que otros lenguajes como Java o C++. A pesar de estas limitaciones, es posible crear en JavaScript algunos elementos parecidos a las clases, lo que se suele denominar pseudoclase. Los conceptos que se utilizan para simular las clases son las funciones constructoras y el prototipo (prototype) de los objetos, que se muestran con detalle a continuación.

3.5.1 Funciones constructora

Las grandes diferencias entre JavaScript y los lenguajes orientados a objetos se acentúan en lo referente a los constructores. En JavaScript no es necesario definir una clase y una serie de constructores predefinidos. El mecanismo empleado para simular en JavaScript el funcionamiento de los constructores se basa en funciones.

Cuando se crea por ejemplo un nuevo objeto genérico o un objeto array, se utilizan las siguientes instrucciones:

```
var elObjeto = new Object();
var elArray = new Array(5);
```

En realidad, se trata del nombre de una función que se ejecuta para crear el nuevo objeto. Como se trata de funciones, es posible pasarles parámetros durante la creación del objeto. De esta forma, JavaScript utiliza funciones para simular los constructores de objetos y de ahí el nombre de “funciones constructoras”.

La función constructora inicializa las propiedades del objeto creado mediante el uso de la palabra reservada this. Normalmente, al crear un objeto se le pasan al constructor de la clase una serie de valores que se asignan inicialmente a algunas propiedades. En el caso de JavaScript, el concepto es similar, aunque su realización es diferente.

La función constructora puede definir todos los parámetros que necesita para construir los nuevos objetos y posteriormente utilizar esos parámetros para la inicialización de las propiedades.

Normalmente, las funciones constructoras no devuelven ningún valor y se limitan a definir las propiedades y los métodos del nuevo objeto.

3.5.2 Prototype

Esta técnica de incluir los métodos de los objetos como funciones dentro de la propia función constructora, funciona correctamente y se puede utilizar para simular clases de cualquier complejidad. Sin embargo, tiene un gran inconveniente que la hace poco aconsejable.

JavaScript incluye una propiedad que no está presente en otros lenguajes de programación y que soluciona este inconveniente. La propiedad se conoce como prototype y es una de las características más poderosas de JavaScript, todos los objetos incluyen una referencia interna a otro objeto. Cualquier propiedad o método que contenga el objeto prototipo, está presente de forma automática en el objeto original. El concepto equivalente en los lenguajes orientados a objetos es que cualquier objeto creado en JavaScript hereda todas las propiedades y métodos de otro objeto llamado prototype. Cada tipo de objeto diferente hereda de un objeto prototype diferente.

Sin utilizar palabras técnicas, se puede decir que el prototype es el molde con el que se fabrica cada objeto de ese tipo. Si se modifica el molde o se le añaden nuevas características, todos los objetos fabricados con ese molde tendrán esas características.

Así, los métodos de los objetos son ideales para incluirlos en el prototype de un objeto. Normalmente, los métodos no varían de un objeto a otro, por lo que se puede evitar el problema de rendimiento comentado anteriormente añadiendo los métodos al prototipo a partir del cual se crean los objetos.

3.6 String

Métodos Públicos

3.6.1 Eliminar espacios en blanco a ambos lados.

trim() : String

Elimina espacios de cualquier cadena, dejando espacios dentro de la cadena intacta.

Muestra una forma en la que se añade los métodos del objeto anterior a su prototipo.

Ejemplo:

```
var s = '  foo bar  ';
alert('-' + s + '-');          //alerta "- foo bar -"
alert('-' + s.trim() + '-');  //alerta "-foo bar-"
```

3.6.2 Eliminar espacios en blanco por la izquierda.

ltrim() : String

Elimina espacios de cualquier cadena por la izquierda dejando espacios dentro de la cadena intacta.

Muestra una forma en la que se añade los métodos del objeto anterior a su prototipo

Ejemplo:

```
var s = '  foo bar  ';
alert('-' + s + '-');          //alerta "- foo bar -"
alert('-' + s.trim() + '-');  //alerta "-foo bar -"
```

Retorna:

- String. La cadena sin espacios a la izquierda.

3.6.3 Eliminar espacios en blanco por la derecha.

rtrim() : String

Elimina espacios de cualquier cadena por la derecha dejando espacios dentro de la cadena intacta.

Muestra una forma en la que se añade los métodos del objeto anterior a su prototipo.

Ejemplo:

```
var s = '  foo bar  ';
alert('-' + s + '-');          //alerta "- foo bar -"
alert('-' + s.rtrim() + '-');  //alerta "- foo bar-"
```

Retorna:

- String. La cadena sin espacios a la derecha.

3.6.4 Eliminar todos los espacios en blanco.

quitaEspacios(String cadena) : String

Elimina todos los espacios de cualquier cadena.

Ejemplo:

```
var s = String.quitaespacios(' 123  5 0 ');
//s ahora contiene la cadena: '12350'
```

Parámetros:

- cadena: String. La cadena original.

Retorna:

- String. La cadena sin espacios.

3.6.5 Rellena una cadena por la izquierda

rightPad(String cadena, Number size, [String char]) : String

<static> Rellena el lado derecho de una cadena con un carácter especificado. Esto es especialmente muy apropiado para normalizar cadenas de número y de fecha.

Ejemplo:

var s = String.rightPad(*'123'*, 5, *'0'*);

//s ahora contiene la cadena: *'12300'*

Parámetros:

- cadena: String. La cadena original.
- size: Number. La longitud total de la cadena de salida
- char: String. (opcional) El carácter con el cual rellena la cadena original (por defecto si la cadena es vacía " ")

Retorna:

- String. La cadena completada.

3.6.6 Devolver cadena después

get_cad_affter(String cadena, String char, Number size) : String

<static> Obtiene de la cadena original una subcadena a partir del carácter especificado, con un valor de ocurrencia del carácter, hacia el final de la cadena, en caso de no encontrarlo devuelve la cadena vacía “ ”.

Ejemplo:

var s = String.get_cad_affter (*'1234'*, 3, *'1'*);

//s ahora contiene la cadena: *'4'*

Parámetros:

- cadena: String. La cadena original.
- char: String. El carácter con el cual se busca en la cadena original
- size: Number. La ocurrencia del carácter buscado.

Retorna:

- String. La subcadena.

3.6.7 Devolver cadena antes

get_cad_before(String cadena, String char, Number size) : String

<static> Obtiene de la cadena original una subcadena a partir del carácter especificado, con un valor de ocurrencia del carácter, hacia el principio de la cadena, en caso de no encontrarlo devuelve la cadena vacía “ ”.

Ejemplo:

var s = String.get_cad_affter (*'1234'*, 3, *'1'*);

//s ahora contiene la cadena: *'12'*

Parámetros:

- cadena: String. La cadena original.
- char: String. El carácter con el cual se busca en la cadena original
- size: Number. La ocurrencia del carácter buscado.

Retorna:

- String. La subcadena.

3.6.8 Validar compuesto

valida_compuesto (String cadena, Number limite, String precede): String

Esta función divide la cadena adicionándole ceros hasta el limite y valida que solo sean números, después de los números vendría el slash seguido del año, además se le puede incluir un precede que se pondría delante de la cadena, el resultado seria el precede (si lo lleva), la cadena de números, el slash y el año.

Ejemplo: (DP-0012/2002)

Parámetros:

- cadena: String. La cadena original.
- limite: Number. El limite de la parte que precede a los últimos 4 caracteres.
- precede: String. Parte que precede a la cadena.

Retorna:

- String. La cadena con la configuración de los números antes del slash y después de este el valor del año y en caso de contener la parte que los precede se coloca delante de la cadena formada.

3.6 Number

Métodos Públicos

3.6.1 Completar número con dos decimales.

completar_num_dosdec (Number number): Number

Completa el número dos decimales. Para ello pregunta por la posición del '.' y en caso de no encontrarlo le agrega al monto ".00", si lo encuentra separa la parte entera y la decimal, si la parte entera es vacía le agrega un 0, en caso que la longitud de la parte decimal sea 3 devuelve un error, si es la longitud es 1 le agrega un 0 al final (por defecto si el campo esta vacío " " se completa con 0.00)
Esto es especialmente muy apropiado para validar que el monto de dinero entrado sea válido.
Ejemplo:

var s = Number.completar_num_dosdec(*'123'*);

//s ahora contiene el número: *'123.00'*

Parámetros:

- number: Number. La cadena original.

Retorna:

- Number. El número completado con dos lugares decimales.

3.6.2 Validar número longitud

val_num(Number number, Number size, Number dec) : Boolean

Obtiene el número hasta los decimales. En caso de no tener números después del punto o con uno solo también es válido.
Ejemplo:

var s = Number.val_num (*'12'*, 4, *'2'*);

//s ahora contiene la cadena: *'12'*

Parámetros:

- number: Number. La cadena original.
- size: Number. La longitud total de la cadena de salida.
- dec: Number. Los lugares decimales.

Retorna:

- Boolean. El número cumple o no con las condiciones.

3.6.3 Validar fecha de Carnet de Identidad

fecha_CI(Number number) : Boolean

Valida la fecha del Carnet de Identidad este en el formato correcto. Los dos primeros caracteres deben cumplir con el año, los caracteres del tercero al cuarto cumpla con los meses del año, y los caracteres del quinto al sexto cumplan con los días que se dispone en ese mes.

Ejemplo:

var s = Number.fechaCI (85012620027);

//s ahora vale: *'true"*

Parámetros:

- number: Number. La cadena original.

Retorna:

- Boolean. El número de C.I. cumple o no con las condiciones de la fecha.

3.6.4 Validar Carnet de Identidad

validaCI(Number number) : Boolean

Valida que el numero de Carnet de Identidad este en el formato correcto. La longitud del número es de 11 caracteres. Verifica el sexo de la persona en su penúltima posición, en caso de ser masculino retorna M de lo contrario F. La fecha verificada con la llamada a la función fechaCI.

Ejemplo:

var s = Number.validaCI (85012620027);

//s ahora vale: *'true'*

Parámetros:

- number: Number. La cadena original.

Retorna:

- Boolean. El número de C.I. cumple o no con las condiciones.

3.8 Date

Métodos Públicos

3.8.1 Comparar fecha

comparar_fecha (String fecha1, String fecha2): Number

Compara la fecha1 con la fecha2.

Ejemplo:

var s = Date.comparar_fecha(*'20/03/2008'*,*'20/03/08'*);

//s ahora vale: *'0'*

Parámetros:

- fecha1: String. La fecha de inicio.
- fecha2: String. La fecha de fin.

Retorna:

- Number. Si el número retornado es mayor que 0 la fecha1 mayor que la 2da, si el número es menor que 0 la fecha 1 menor que la 2da, si son iguales el valor retornado es 0.

3.8.2 Devuelve la fecha en un formato.

get_fecha (): String

Obtiene la fecha actual en el formato de 'd/m/Y'.

Ejemplo:

var s = Date.get_fecha();

//s ahora vale la fecha actual.

Parámetros:

- None

Retorna:

- String. Fecha en el formato 'd/m/Y'

3.8.3 Validar fecha mayor al día de hoy

fecha_mayor_hoy (String fecha): Boolean

Compara el parámetro de fecha con la fecha actual.

Ejemplo:

var s = Date.fecha_mayor_hoy(*'20/03/2008'*);

//s vale el valor de la fecha si es mayor que la actual.

Parámetros:

- fecha1: String. La fecha de inicio.

Retorna:

- Boolean. La fecha en caso de ser mayor que la actual.

3.8.4 Validar fecha mayor o igual que el día de hoy

En las propiedades del campo de fecha se encuentra la de la fecha sea mayor igual a la actual.

minValue : new Date, pero se necesitaba una llamada a una función que se ejecute.

fecha_mayor_igual_hoy (String fecha): Boolean

Compara la fecha con la fecha actual.

Ejemplo:

var s = Date.fecha_mayor_igual_hoy(*'20/03/2008'*);

//s vale el valor de la fecha si es mayor o igual que la actual.

Parámetros:

- fecha1: String. La fecha de inicio.

Retorna:

- Boolean. La fecha en caso de ser mayor o igual que la actual.

3.8.5 Validar fecha menor o igual que el día de hoy

En las propiedades del campo de fecha se encuentra la de la fecha sea menor igual a la actual.

maxValue : new Date, pero se necesitaba una llamada a una función que se ejecute.

fecha_menor_igual_hoy (String fecha): Boolean

Compara la fecha con la fecha actual.

Ejemplo:

var s = Date.fecha_menor_igual_hoy(*'20/03/2008'*);

//s vale el valor de la fecha si es menor o igual que la actual.

Parámetros:

- fecha1: String. La fecha de inicio.

Retorna:

- Boolean. La fecha en caso de ser menor o igual que la actual.

3.8.6 Validar que la fecha inicio sea mayor igual que la fecha fin

fecha_inicio_fin (String fecha1, String fecha2): Boolean

Compara la fecha inicio con la fecha fin.

Ejemplo:

var s = Date.fecha_inicio_fin(*'20/03/2008','21,03,2008'*);

//s ahora vale: true

Parámetros:

- fecha1: String. La fecha inicio.
- fecha2: String. La fecha fin.

Retorna:

- Boolean. La fecha de inicio es menor que la fecha fin.

3.8.7 Validar fecha inicio

valida_finicio (String fecha1, String fecha2, String fecha3, Integer opción): Boolean

1. Compara la fecha1 si es diferente a la fecha2, si la opción es 1, 2, 3, 4 o 6, y la fecha1 actual es mayor que la actual.

2. Compara la fecha1 si es diferente a la fecha2, si la opción es 5, si la fecha1 es mayor que la actual y mayor que la fecha 3.

Esto es especialmente muy apropiado para la fecha de un artículo que entra en vigencia.

Ejemplo:

var s = Date.valida_finicio(*'05/05/2008'*,*'10/04/2008',30/04/2008",5*);

//s ahora vale: *'05/04/2008'*

Parámetros:

- fecha1: String. La fecha de inicio.
- fecha2: String. La fecha de inicio del anterior
- fecha3: String. La fecha de fin del anterior.
- opción: Number. El número de la opción.

Retorna:

- Boolean. Devuelve la fecha en caso de cumplir con todas las condiciones.

3.8.8 Validar fecha fin

valida_ffin (String fecha1, String fecha2, String fecha3, Integer opción): Boolean

1. Si la fecha 1 no es vacía y la opción es 1, 2, 3, 5, o 6.

2. Compara la fecha1 si es mayor que la actual, mayor que la fecha 2 y si a opción es 4, compara si fecha1 es menor que fecha 3.

Esto es especialmente muy apropiado para la fecha de un artículo que entra en vigencia.

Ejemplo:

var s = Date.valida_finicio(*'20/03/2008'*,*'20/03/08',",3*);

//s ahora vale

Parámetros:

- fecha1: String. La fecha de inicio.
- fecha2: String. La fecha de inicio del próximo.
- fecha3: String. La fecha de fin del próximo.
- opción: Number. El número de la opción.

Retorna:

- Boolean. Devuelve la fecha en caso de cumplir con todas las condiciones.

3.9 Conclusiones

En los sistemas web la validación de datos, es de vital importancia. Tan sensible es el manejo de la información que el mínimo error puede causar una catástrofe de dimensiones insospechadas. Así de crítico es el crecimiento de los datos en sistemas y más aún, en el caso de grandes sistemas, donde se debe ingresar información de una cantidad considerable de documentos de carácter confidencial.

Un rol esencial en esta tarea la tienen aquellas personas responsables de facilitar la información. Aquí la rigurosidad y conciencia de la magnitud de la actividad permite ingresar de forma segura y sólida los datos que se dispondrá para los funcionarios. Es decir, el usuario que está al otro lado de la pantalla confía plenamente que la información entregada es la correcta, y por otro lado quien la entregó está seguro de haber puesto a disposición datos en forma correcta al usuario final, según su área o perfil.

Finalmente el cumplimiento de una metodología de trabajo adecuado y responsable permite la entrada de datos de forma segura en el sistema y con ello validar ante los usuarios los datos disponibles, convirtiéndolo en un sistema potente para los funcionarios de la institución.

CONCLUSIONES

La concepción, diseño e implantación de la librería ExtJS constituye un serio aporte en el desarrollo y creación de sistemas más ágiles e interactivos. Mediante su exitosa introducción en las condiciones concretas para la Aduana General de la República se han creado condiciones materiales y subjetivas que garantizan el futuro perfeccionamiento en el desarrollo del sistema. Durante el desarrollo de la presente investigación se han empleado tecnologías novedosas y de gran alcance en la Informática facilitando el desarrollo. Esto implica que necesariamente se utilicen nuevas herramientas y tecnologías, de las cuales se necesita información. Es debido a esto que para la realización y desarrollo de este trabajo se buscó, investigó y agrupó, un cúmulo de información referente a distintos materiales de soporte (como es el caso de las librerías JavaScript) para su posterior utilización por parte de la Aduana General de la República (AGR); y que además se adapten a las condiciones concretas de nuestro país. Es por lo que se realiza una extensa investigación del framework ExtJS con el objetivo general definido de:

Proponer una solución que permita evitar la entrada de datos erróneos en la aplicación mediante las implementaciones de las validaciones necesarias, integrándoselas a al framework JavaScript ExtJS.

Por lo cual se pueden establecer las siguientes conclusiones:

Se llevo a cabo un proceso investigativo extenso con la finalidad de proporcionarle al usuario la mejor opción para el desarrollo ágil de su trabajo así como informaciones de validaciones precisas.

La librería seleccionada cumple con los requisitos mínimos que exigía el proyecto para su posterior utilización como son: flexibilidad, usabilidad, facilidad de uso y adaptación a cambios que se pudieran producir en el futuro, así como su posterior actualización.

RECOMENDACIONES

- Desarrollar un framework de validaciones propio para la AGR, ligero y escalable a cualquier cambio que se necesite.
- Fácil de adaptar a cualquier otro framework y sistemas que se desarrollen.

BIBLIOGRAFÍA

(1) Johnson, Ralph E. and Brian Foote (1988). Designing Reusable Classes. Journal of Object-Oriented Programming.

(2) Johnson, Ralph E (1997). Components, Frameworks, Patterns. Proceedings of the 1997 symposium on Software reusability.

(3) Gamma, Erich, Richard Helm, Ralph Johnson (1995). et al. Design Patterns. Elements of Reusable Object-Oriented Software. Addison Wesley.

(4) Sitio oficial de Symfony http://www.symfony-project.com.

(5) Potencier, Fabien y Zaninotto, François (2007). The Definitive Guide to Symfony. 1, 12-14. Recuperado 15 de Diciembre de 2007, de http://www.librosweb.es/symfony/capitulo1/symfony_en_pocas_palabras.html

(6) Bushmann, Frank, Regine Meunier, Hans Rohnert (1996), et al. Pattern-Oriented Software Architecture: A System of Patterns. England, John Wiley and Sons.

(7) Librería de funciones Javascript. Recuperado 5 de Noviembre de 2007 de http://www.desarrolloweb.com/articulos/705.php

(8) Eguíluz Pérez, Javier. Introducción a JavaScript, 2007. Recuperado 15 de Noviembre de 2007, de http://www.librosweb.es/javascript/index.html

(9) Sitio oficial http://www.prototypejs.org/. Documentación Prototype JavaScript Framework (2008). Recuperado 25 de noviembre de 2007 de http://www.prototypejs.org/api

(10) Sitio oficial http://www.dojotoolkit.org. Documentación Dojo JavaScript Toolkit (2008). Recuperado 27 de noviembre de 2007 de http://dojotoolkit.org/docs

(11) Sitio oficial http://jquery.com. Documentación JQuery JavaScript Library (2008). Recuperado 29 de noviembre de 2007 de http://docs.jquery.com/Main_Page

(12) Sitio oficial http://developer.yahoo.com/yui. Documentación Yahoo! User Interface Library (YUI) (2008). Recuperado de http://developer.yahoo.com/yui/docs

(13) Sitio oficial http://extjs.com. Documentación Ext JS JavaScript Library (2008). Recuperado de http://extjs.com/deploy/dev/docs

(14) Licencia de Dojo Toolkit. Recuperado de http://dojotoolkit.org/license

(15) Licencia de la librería JavaScript JQuery. Recuperado de http://docs.jquery.com/Licensing

(16) Licencia de Yahoo! UI Library. Recuperado de http://developer.yahoo.com/yui/license.html

(17) Licencia de Ext JS and Ext GWT. Recuperado de http://extjs.com/products/license.php

(18) Los mejores 12 FrameWorks JavaScript. Recuperado 5 de Enero de 2007 de http://www.xperimentos.com/2007/09/04/los-mejores-12-frameworks-javascript

(19) Documentación científico-técnica de EzWeb runtime. Recuperado 14 de diciembre de 2007 de http://forge.morfeo-project.org/wiki/index.php/D.3.4_Documentaci%C3%B3n_cient%C3%ADfico-t%C3%A9cnica_de_EzWeb_runtime

(20) Sitio oficial de Aptana http://www.aptana.com

(21) Aptana, IDE para aplicaciones AJAX. Recuperado 13 de Noviembre de 2007, de http://www.genbeta.com/2006/07/26-aptana-ide-para-aplicaciones-ajax

(22) Mozilla Foundation, Firefox 2 (2007). Recuperado 13 de Noviembre de 2007, de http://www.mozilla-europe.org/es/products/firefox/features

(23) Validación de formularios. Symfony, la Guía definitiva. Recuperado 20 de enero de 2008, de http://librosweb.es/symfony/capitulo10/validacion_de_formularios.html

GLOSARIO DE TÉRMINOS

API: Acrónimo de "Application Programming Interface" (en español Interfaz de Programación de Aplicaciones) es del conjunto de funciones y procedimientos (o métodos si se refiere a programación orientada a objetos) que ofrece cierta librería para ser utilizado por otro software como una capa de abstracción.

Widgets: Es una pequeña aplicación o programa, usualmente presentado en archivos o ficheros pequeños que son ejecutados por un motor de widgets o Widget Engine. Entre sus objetivos están los de dar fácil acceso a funciones frecuentemente usadas y proveer de información visual. Sin embargo los widgets pueden hacer todo lo que la imaginación desee e interactuar con servicios e información distribuida en Internet; pueden ser vistosos relojes en pantalla, notas, calculadoras, calendarios, agendas, juegos, ventanas con información adicional, etc.

DOM: Acrónimo de "Document Object Model" (en español 'Modelo de Objetos de Documento'), frecuentemente abreviado DOM, es una forma de representar los elementos de un documento estructurado (tal como una página Web HTML o un documento XML) como objetos que tienen sus propios métodos y propiedades. El responsable del DOM es el World Wide Web Consortium (W3C).

JSON: Acrónimo de "JavaScript Object Notation", es un formato ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML. La simplicidad de JSON ha dado lugar a la generalización de su uso, especialmente como alternativa a XML en AJAX. Una de las supuestas ventajas de JSON sobre XML como formato de intercambio de datos en este contexto es que es mucho más sencillo escribir un analizador semántico de JSON.

AJAX: Acrónimo de "Asynchronous JavaScript + XML" (en español JavaScript asíncrono y XML), no es una tecnología. Es realmente una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Éstas se ejecutan en el cliente, es decir, en el navegador de los usuarios y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la misma.

XML: Acrónimo de “Extensible Markup Language” (en español Lenguaje de marcas extensible), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML.

DHTML: Acrónimo de “Dynamic HTML” (en español HTML Dinámico) Designa el conjunto de técnicas que permiten crear sitios web interactivos utilizando una combinación de lenguaje HTML estático, un lenguaje interpretado en el lado del cliente (como JavaScript), el lenguaje de hojas de estilo en cascada (CSS) y la jerarquía de objetos de un DOM.

XPath: Acrónimo de “XML Path Language”. Es un lenguaje que permite construir expresiones que recorren y procesan un documento XML. La idea es parecida a las expresiones regulares para seleccionar partes de un texto sin atributos (texto plano). XPath permite buscar y seleccionar teniendo en cuenta la estructura jerárquica del XML. XPath fue creado para su uso en el estándar XSLT, en el que se usa para seleccionar y examinar la estructura del documento de entrada de la transformación.

1