15. (José Sempere) fgutierrez

Ejemplo de comunicación bidireccional (consultas, altas, modificaciones y bajas de registros, replicación de tablas) entre aplicaciones v6 y v7. Coexistencia de aplicaciones de ambas plataformas.

Existen varias formas de mantener comunicación bidireccional entre aplicaciones de Velneo 6.x y Velneo V7:

- Función remota en 6.x contra V7: Por medio de funciones remotas con la librería vRemoteFunctionV7.dll que se suministra con Velneo vClient V7.
  http://velneo.es/ejecucion-desde-velneo-6x-de-funciones-remotas-de-velneo-v7/
  http://velneo.es/como-hacer-y-usar-funciones-remotas-v7/
  http://velneo.es/traspaso-datos-6x-v7-con-vremotefunctionv7-dll/

- Protocolo TCP: Totalmente bidireccional, podemos hacer que tanto un cliente de 6.x como un cliente o el servidor de Velneo V7 puedan actuar como servidores de información.

- Servidor Web: Tanto Velneo V7 como Velneo 6.x nos permiten por medio del servidor web compartir información de forma sencilla.

- Otros: Ficheros planos, etc.

Notas:

Velneo V7 permite puertos distintos de 690, por lo que pueden coexistir dos servidores de Velneo en la misma máquina.

16. (Just Serrainat) fgutierrez

Cuando queremos enviar un VIN con actualizaciones de una aplicación. Existe una manera de automatizar el borrado de las instancias antiguas antes de instalar las nuevas. Sería en el caso de que se hayan añadido proyectos nuevos por ejemplo.

- Actualmente .vin está pensado para altas.
- Para simplificar esta tarea podemos usar *Importar componentes*.
- Si hay proyectos nuevos no es necesario borrar las antiguas, bastará crear las nuevas instancias y tras reiniciar la solución, conectar las instancias que se encuentren pendientes de conectar.
- Estamos diseñando un API para las instalaciones.

17. (María del Rocío) jarboleya

¿Cómo utilizar el campo de hermano contiguo?

1. ¿Qué es un hermano contiguo?

2. ¿Cómo se declaran?
    a. Partes comunes (Hermanos por parte de madre). Índices con varias partes.
    b. No copiar y pegar este tipo de campo entre tablas.

3. Uso:
    a. Botón en formulario para moverse a hermano. (Ver formulario de clientes).
    b. Botón en formulario para moverse a hermano contiguo por parte de madre (Ver formulario de pedidos).
    c. Contenido inicial de un campo (Ver tabla líneas, campo CAN_PED_ACU).
    d. Leer ficha de maestro en un proceso y arrastrado a stock mediante triggers y procesos (Ver proceso CAL_CAN_ACU).
    e. Con el API podemos simular campos hermano sin necesidad de crear el campo:
        i. vBase_Usuarios. Ver carpeta utilidades -> Hermano contiguo.
        ii. Ver ejemplo en vHelpDesk 7.14 -> Estado -> Pendientes -> Hermano.
    f. Tener en cuenta su rendimiento, por ejemplo al usarlo en rejillas o formularios.

18. (Pablo Navarrete) fgutierrez

1. Al crear una solución, ¿en qué caso concreto se utiliza el desmarcar la opción de compartida?

2. ¿Cómo podemos utilizar los identificadores que se generan para cada uno de los proyectos?

3.Usos de Hojas de estilos.

1. Soluciones independiente que no queramos sea compartida por otras (excepción).

2. Los identificadores de proyecto deben evitarse en la medida de lo posible. Recomendamos el uso de alias donde sea posible. Únicamente queda algún sitio donde se usa el identificador del fichero como en la inclusión de ficheros javascript.

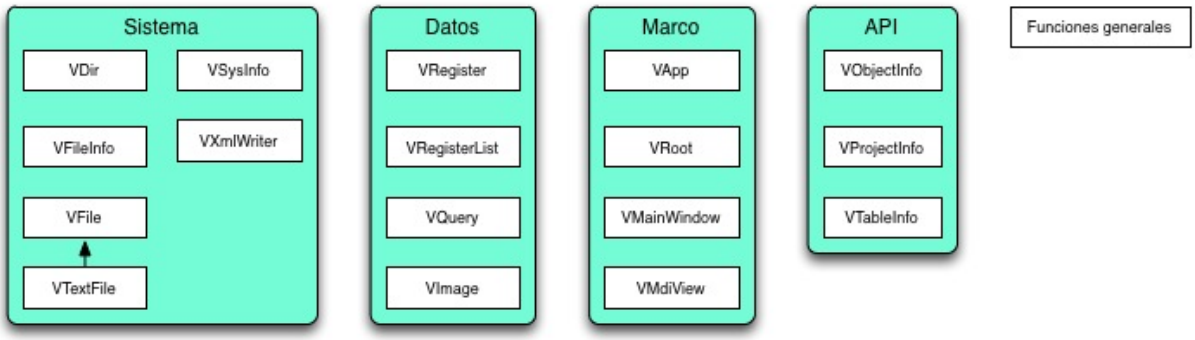
- Ejemplo de uso de CSS:
  - Cascada, autoexec y objetos incluidos o llamados.
  - Cada objeto puede tener sus CSS aplicadas en el evento post inicializado.

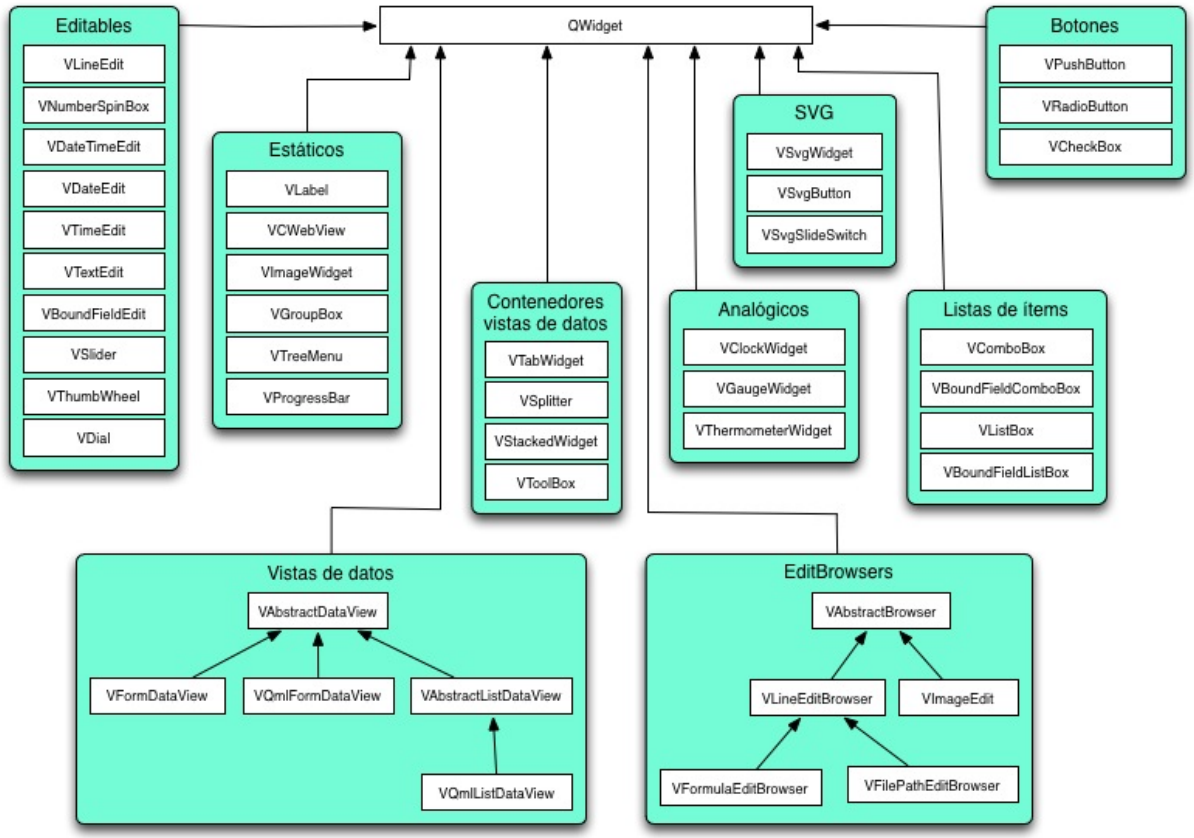
http://velneo.es/info_v7_714_es/velneo_vdevelop_v7/proyectos_objetos_y_editores/proceso/comandos/interfaz/control/interfaz_establecer_hoja_de_estilo_css_filescript/

20. (Joel Cabrera) jarboleya

¿Que podemos llegar a hacer con Javascript en Velneo?

1. Lo primero que tenemos que comprender es que más que de JavaScript tenemos que hablar del API de Velneo V7.
    a. Una librería de clases que dan acceso a múltiples funcionalidades de la plataforma a bajo nivel.
    b. Estas clases no son específicas de un lenguaje concreto y se usan tanto desde JavaScript como desde QML.





2. Se pueden hacer muchas, muchas, muchas... cosas.
    a. Un número importante de funcionalidad de JavaScript es equivalente a lo que se puede hacer con V7:
        i. Manejar variables globales.
        ii. Variables locales.
        iii. Búsquedas.
        iv. Transacciones: altas, bajas, modificaciones, etc.
        v. Ejecutar procesos.
        vi. Operciones de interfaz, mostrar vistas.

3. La abstracción y el acceso a bajo nivel es lo verdaderamente importante.
    a. Leer los proyectos de datos sus tablas, campos, índices, etc.
    b. En ejecución obtener un formulario, una pestaña, un control, una vista, etc.

4. ¿Qué podemos hacer?
    a. Ejemplo Mi Gestión o vAgenda para ver utilidades incluidas en vBase para Filtrado, Combinar listas o Exportación ASCII.

5. Procesos V7 y los procesos JavaScript pueden interactuar.
    a. Desde un proceso V7 podemos ejecutar un proceso JavaScript.
    b. Desde un proceso JavaScript podemos ejecutar un proceso V7.
    c. Los procesos V7 y JavaScript pueden pasarse información a través de:
        i. Tablas en disco o memoria.
        ii. Variables globales.
        iii. Variables locales.

6. Adicionalmente, podemos encontrar infinidad de código JavaScript funcional para resolver muchísimas soluciones. Este código podemos combinarlo con el API de Velneo V7 conseguir resolver necesidades que no existen por defecto en la plataforma.
    a. Ejemplo de edit de etiquetas.

26. (Pedro Porlan) jarboleya

Acceso desde distintos entornos (manejadores de evento, funciones, TCP...) Javascript a los objetos theRoot, theApp...

1. Cuando ejecutamos un proceso o manejador de evento JavaScript Velneo alimenta al motor de ejecución de JavaScript con objetos predefinidos.
2. En función del entorno de ejecución dispondremos diferentes objetos. Estos objetos son únicos, por eso llevan el prefijo “the”:

    a. **theApp**: Este objeto representa a la aplicación. Dispone de funciones para ver información de sus proyectos, acceso a variables globales, constantes, utilidades de base de datos, disco, etc.

    b. **theMainWindow**: Nos da acceso a las funciones de manejo de la ventana principal de la aplicación.

    c. **theRegister**, **theRegisterIn**, **theRegisterListIn**, **theRegisterListOut**: Representa a un registro o listas de registros.
        i. En una fórmula theRegister representa el registro de entrada.
        ii. En un proceso de origen ficha theRegisterIn representa el registro de entrada.
        iii. En un proceso de destino ficha theRegisterOut representa el registro de salida.
        iv. En un proceso de origen lista theRegisterListIn representa la lista de registros de entrada.
        v. En un proceso de destino lista theRegisterListOut representa la lista de registros de salida.

    d. **theRoot**: Se corresponde con el objeto principal en ejecución. Dispone de funciones para ver o modificar las variables locales y de sistema del objeto, obtener información del objeto (api), transacciones, etc. Veamos algunos ejemplos:
        i. Tenemos un formulario con un manejador de evento en lenguaje JavaScript: el objeto theRoot del manejador de evento representa al ejecutor del formulario.
        ii. Ejecutamos un proceso JavaScript con una acción: el objeto theRoot del proceso representa al ejecutor del proceso en V7.
        iii. Si el objeto v7 es una vista de datos (rejilla, formulario…) estarán

disponibles en theRoot las funciones de manejo de interfaz como obtener su vista de datos con la función dataView(), pudiéndose usar las funciones del objeto widget que devuelve.

iv. También se pueden obtener otros VRoot, como el de una VMdiView con su funcion root().

Pol. Ind. A Granxa, Edificio vCenter
Rúa D, esquina paralela 3
36400 Porriño (Pontevedra)
España

Telf. (+34) 902 02 02 61
velneo@velneo.com
velneo.es

27. (Rafael Cortés Jurado) fgutierrez

¿Es posible automatizar la gestión de usuarios de velneo, de forma que si un usuario pierde su contraseña, en un entorno cloud, pueda recibir otra de forma "desatendida"?-¿Puedo tener un usuario, que administre un grupo de usuarios, sin ser administrador?

Por medio del Cloud API es posible programar la recuperación de la contraseña de un usuario para servidores en la nube.

- Entre las funciones de Cloud API se encuentra la posibilidad de modificar un usuario y configurar una nueva contraseña.
- Podemos acceder por REST o bien por funciones remotas contra el servidor de Cloud API.
- Iniciamos sesión
- Establecemos credenciales si vamos a realizar operaciones de usuarios/grupos/instancias en el servidor
- Realizamos las operaciones necesarias durante la sesión

Los usuarios supervisores que acceden al servidor son los únicos que pueden administrar usuarios.

Componente
http://velneo.es/velneo-open-app/componente-velneo-vcloudapi/

Tutor
http://velneo.es/velneo-open-app/tutor-velneo-vcloudapi/

Documentación de API
http://velneo.es/documentacion-velneo-cloud-api-v1-2/

28. (Manuel Blázquez) fgutierrez

Me gustaría saber como personalizar la pantalla de login del vClient con QML, ya que lo que se explica en el blog y en el foro no me funciona, no encuentro los ficheros qml de los que se habla.

Manual
http://velneo.zendesk.com/attachments/token/dblbcwirbfwmbgy/?name=Documentaci%C3%B3n+de+Login+QML.pdf

Ejemplos
http://velneo.zendesk.com/attachments/token/tkcl8klowigtxmw/?name=LoginQML_ejemplos.zip

(Ejemplo)

31. (Gustavo Sandoval Aipuru) fgutierrez

Cambié el identificador de una tabla y tengo que renombrar .dat y .idx y el vAdmin no me deja Como se hace? Para cambiar el tipo, creo una tabla nueva Y si tengo datos?

- Para renombrar, si es en la nube, tendrás que hacerlo con un proceso en otra solución de "mantenimiento" que actúe sobre los ficheros.
- Otra opción, es un proyecto de datos con la nueva tabla y un proceso de importación. En la siguiente versión, ya se puede borrar la tabla original.
- Deberás crearla nueva con el mismo identificador de tabla y usando el traspaso de campos mover los datos antiguos que tengan diferente identificador. Puede que requiera un proceso de migración posterior que adecue los datos. Recomendamos probarlo previamente en un servidor de pruebas.

32. (Vicente Linares Rams) jarboleya

Subindexaciones

Os recomiendo la lectura de estos 3 artículos que lo explican en profundidad:

http://velneo.es/como-funciona-la-subindexacion-1/

http://velneo.es/como-funciona-la-subindexacion-2/

http://velneo.es/como-funciona-la-subindexacion-3/

(Ejemplo)

Bases de datos externas, mysql

Tutor
http://velneo.es/velneo-open-app/tutor-de-acceso-a-base-de-datos-externas-10/
Ejemplo
http://velneo.es/acceso-a-bases-de-datos-externas/

35. (Esteban Fernández) jarboleya

Procesos segundo plano y tercer plano desde javascript, como? Ficheros js donde se deben ubicar proyecto datos o aplicación? Existe diferencias de rendimiento?

- Procesos en 2º y 3º plano en javascript (7.15):
    - Nueva clase **vObjectInstance**: Clase que representa una instancia de un objeto en ejecución, como por ejemplo una búsqueda (VQuery) o un proceso (VProcess). Esta clase implementa las funciones comunes a todos los objetos como es el uso de las variables locales.
    - Nueva clase **vProcess**: La clase VProcess representa un proceso en ejecución. Nos permite ejecutar procesos. Construimos un objeto VProcess pasándole el objeto VRoot en curso y le indicamos el proceso a ejecutar con la función setProcess. Una vez inicializado el objeto ya podemos usar el resto de funciones (salvo result): le pasamos un registro (setRegisterIn) o una lista de registros (setRegisterListIn) en función del tipo de entrada del proceso y también el contenido a las variables locales del proceso con setVar (si lo requiere) y la ejecutamos con la función exec. Ahora ya podemos recoger los resultados con la función result, que nos devuelve un objeto nulo, un objeto de la clase VRegister o un objeto de la clase VRegisterList en función del tipo de salida del proceso.

```
1  // ---------------------------------------------------------------------------
2  // Ejemplo de ejecución de un proceso de origen ficha persona y destino lista de películas
3  // ---------------------------------------------------------------------------
4
5  importClass("VProcess");
6
7  var proceso = new VProcess(theRoot);
8  proceso.setProcess("CINE/PROCESS1"); // IN: ficha persona. OUT: lista peliculas
9
10 var persona = new VRegister(theRoot);
11 persona.setTable("CINE_DAT/PERSONAS");
12
13 if ( persona.readFirstRegister("ID") )
14 {
15   if ( proceso.setRegisterIn(persona) )
16   {
17     if ( proceso.exec() )
18       theRegisterListOut.append( proceso.result() );
19   }
20 }
```

- Dónde ubicar ficheros js: en el proyecto de más bajo nivel de herencia

- En general no hay diferencias de rendimiento notables respecto a procesos V7. Los tiempos son muy similares y pueden oscilar a favor de uno u otro dependiente de los tipos de instrucción que se ejecutan.
    - Hay una nueva función en JavaScript que permite al programador forzar la ejecución de la cola de eventos del sistema para evitar ralentización en la ejecución que se pueda producir en algún caso muy concreto.

Pol. Ind. A Granxa, Edificio vCenter
Rúa D, esquina paralela 3
36400 Porriño (Pontevedra)
España

Telf. (+34) 902 02 02 61
velneo@velneo.com
velneo.es

38. (Silvia Sánchez Rodríguez) fgutierrez

Con lenguaje v7: descargar campos (cualesquiera de una tabla) a un Excel directamente

- Velneo v2Excel (Open App para Windows)
  http://velneo.es/velneo-open-app/v2excelwin-10/
  http://velneo.es/velneo-open-app/tutor-v2excelwin/

- Generar csv (ImpExpJS)
  http://velneo.es/velneo-open-app/importacion-y-exportacion-dinamica-con-javascript/

- VBScript http://velneo.es/velneo-open-app/vxwinscript/

(Ejemplos)

40. (Juan Valle) fgutierrez

Autoalta de maestro imitar la funcionalidad de la V6

Mediante la señal tecla pulsada guardamos en una variable la información escrita por el usuario.

API de Velneo:
var texto = theRoot.dataView().control("CONTROL_MAESTRO").text;
theRoot.setVar("VARIABLE_MAESTRO", texto);

En la pérdida de foco usamos la información de la variable para buscar el registro o crearlo.

(Ejemplo en Pedidos)

41. (Carlos Moreno) jarboleya

Manejador de objetos

Explicación de manejador de objetos

- Objetos de interfaz: Formularios, Rejillas, Casilleros, etc.
- Objetos sin interfaz: Procesos y Búsquedas (sin formulario)
- Sólo los procesos pueden ejecutarse en diferentes planos.
- Optimizar búsquedas en tercer plano:
    - Deben ser ejecutadas directamente en el servidor si tienen varios componentes.
- Un proceso ejecutado con manejador es como una función vitaminada.

(Ejemplo)

42. (Alfredo Rocha Margain) jarboleya

Me interesa en gran manera conocer más sobre variables

- Arquitectura:
  - Cliente/Servidor.
  - Planos de ejecución.

- Tipos de variables:
  - Globales: Compartidas y accesibles desde todos los planos.
  - Locales: Específicas del objeto.

- Ámbito:
  - Globales:
    - Disco: Siempre compartidas por todos los usuarios de la instancia.
    - Memoria: Dependen del plano en que se use.
  - Locales.
    - Siempre son en memoria.

- Visibilidad:
  - Proceso en 1º plano: Todos comparten las variables en memoria.
  - Proceso en 2º plano: Cada hilo tiene su instancia
  - Proceso en 3º plano: Todos los procesos comparten variables globales en memoria.
  - Manejadores de objeto: Las variables locales son específicas de cada instancia del objeto.
  - Tablas: Las variables locales son visibles tanto en los triggers como en las actualizaciones.
  - Accesibles desde comandos V7 y desde el API.

- Rendimiento:
  - Disco:
    - En el servidor son muy rápidas.
    - En el cliente requieren establecer un socket para garantizar su valor actual.