

3-5 Dic'08

4^{as} Jornadas
Internacionales
gvSIG

Palacio de Congresos de Valencia



Taller para desarrolladores

IV Jornadas gvSIG



Francisco José Peñarrubia
fpenarru@gmail.com

Victor Olaya
volaya@unex.es

César Martínez Izquierdo
volaya@unex.es

Indice



Introducción

Arquitectura interna

Proyectos básicos. Core de gvSIG

Configurar un workspace de trabajo con Eclipse

Andami y el mecanismo de plugin

Ejemplos de extensiones

Hola Mundo

Información personalizada

Conexión a un servicio remoto

Introducción

Aplicación de escritorio

Proyecto escalable desde sus orígenes.

Muchas librerías útiles por separado

Licencia GPL

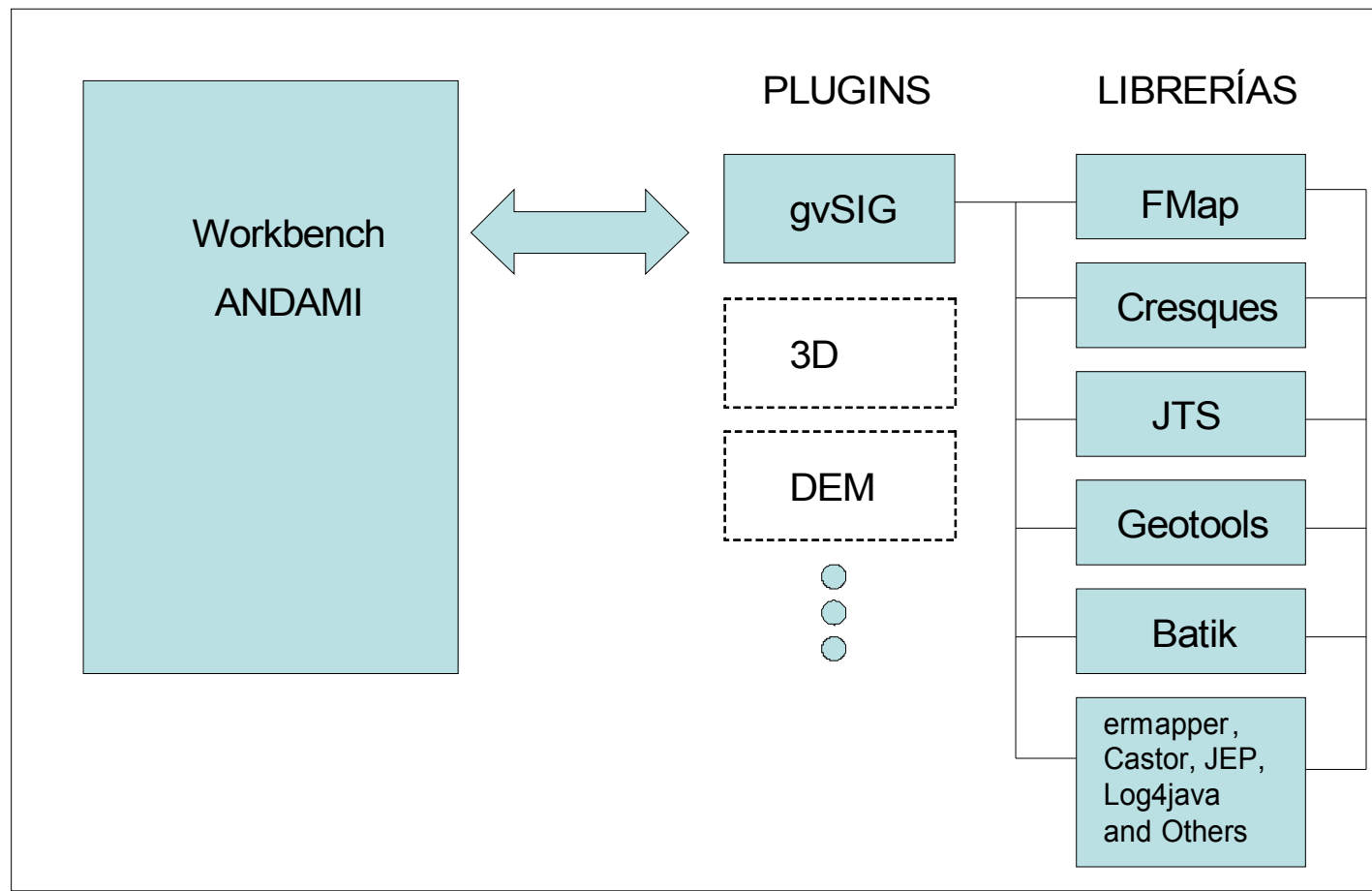
Entorno de desarrollo usado: Eclipse 3.2

Máquina virtual: 1.5 + JAI

Extensiones JNI para otras librerías en C++

Arquitectura Interna

gvSIG



Proyectos básicos

_fwAndami

libCorePlugin

libFMap

libGDBMS

appGvSIG

libCq CMS for Java

extCAD

ExtJDBC

Otros: libDWG, libIverUtiles, libUI, binaries
libRemoteServices, libGPE....

Configurar un workspace de trabajo

Descargamos el código fuente de
<http://www.gvsig.org>

Lo descomprimimos en un directorio cualquiera

En Eclipse ejecutamos *File => Switch Workspace*
(cambiar de workspace)

En el diálogo que aparece seleccionamos el
directorio en el que acabamos de descomprimir las
fuentes

Configurar un workspace de trabajo

gvSIG está compuesto de muchos proyectos (librerías propias)

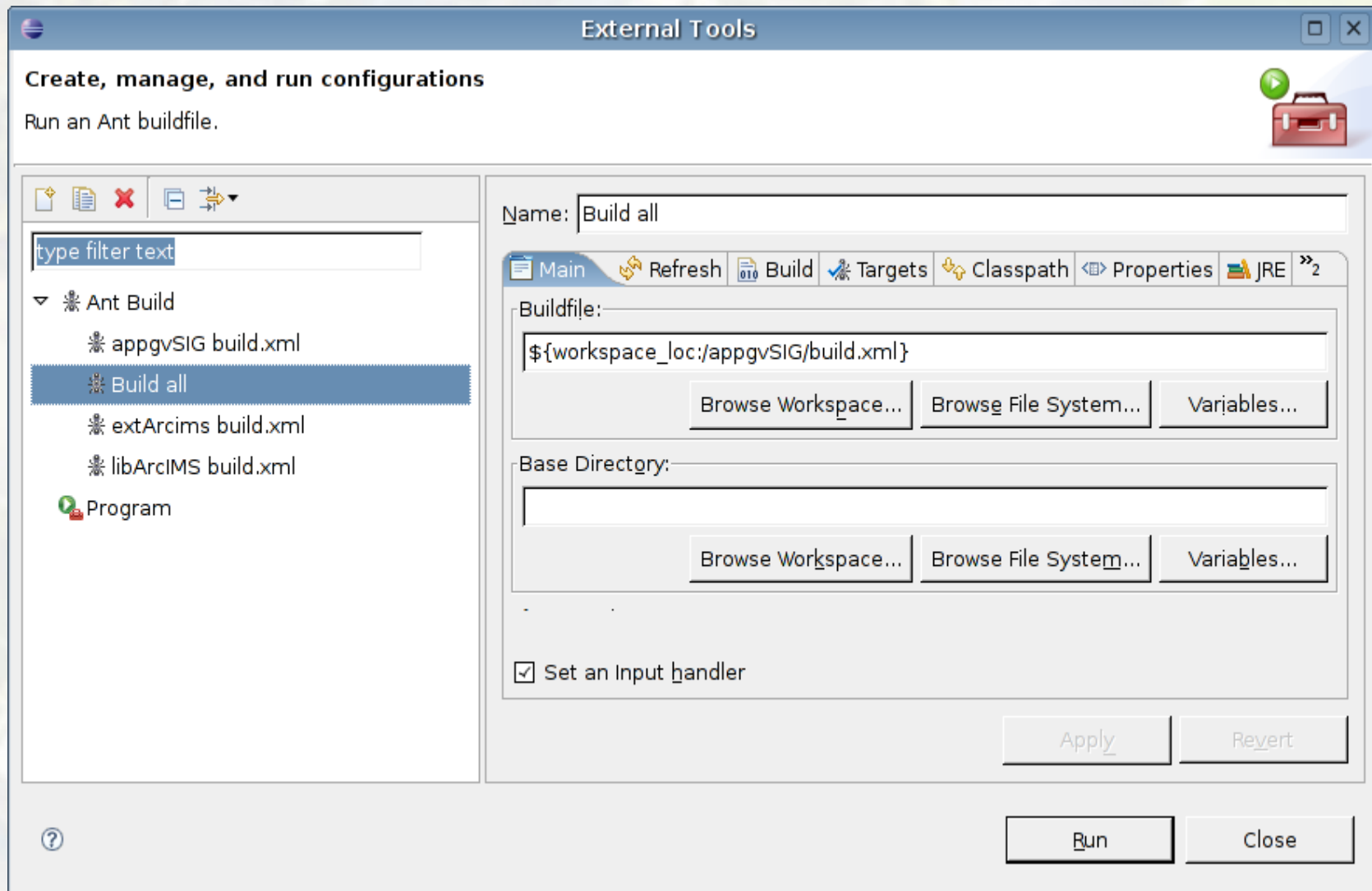
Para compilar gvSIG hay que compilar todos los proyectos que lo componen

Eclipse realiza la compilación (generación de los .class) automáticamente, después debemos ejecutar el fichero build.xml de cada proyecto, que crea los ficheros .jar y los copia al directorio adecuado

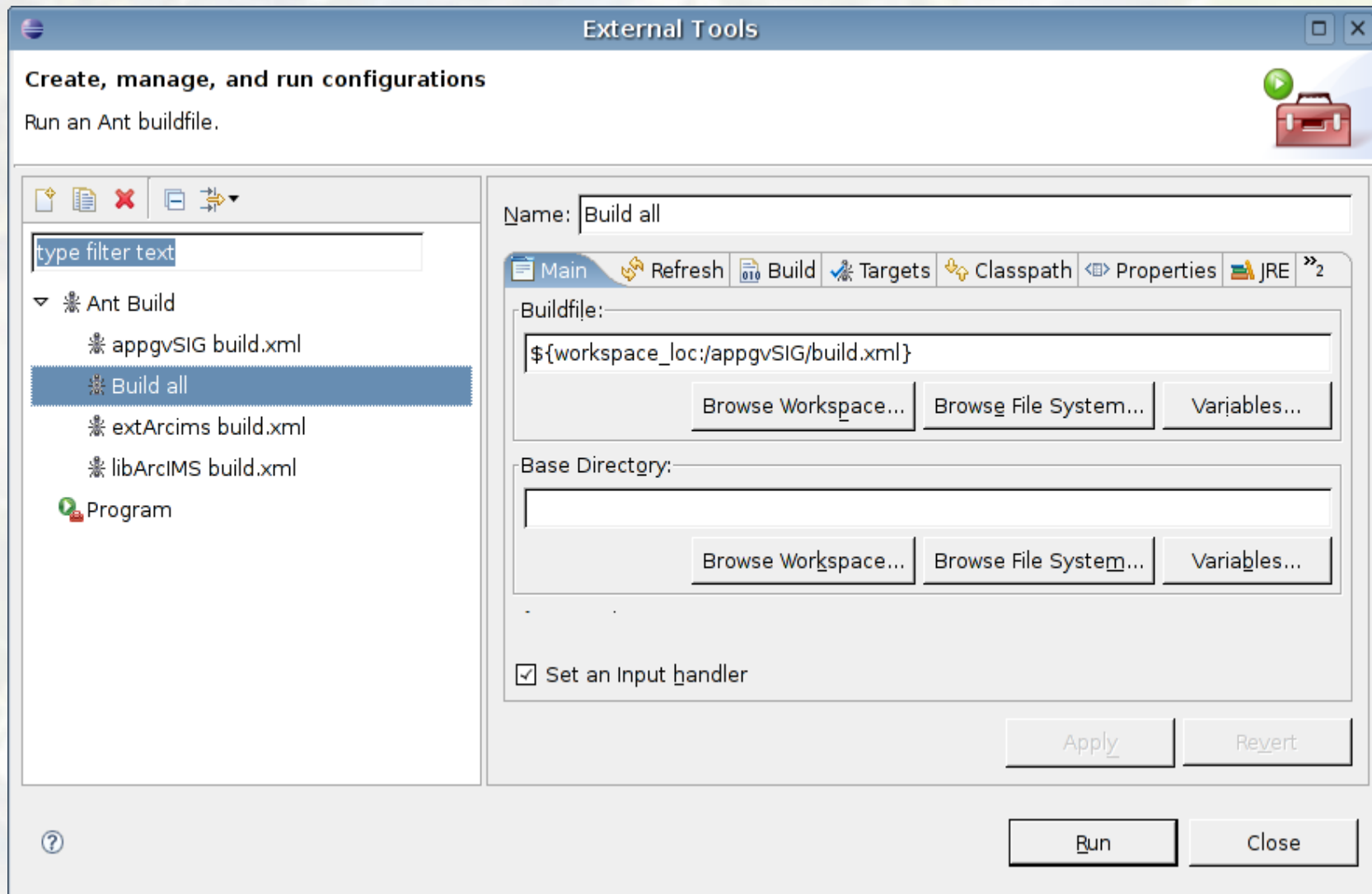
El orden de compilación no es indiferente, ya que unos proyectos necesitan de otros

Para evitar este proceso, existe un método automático

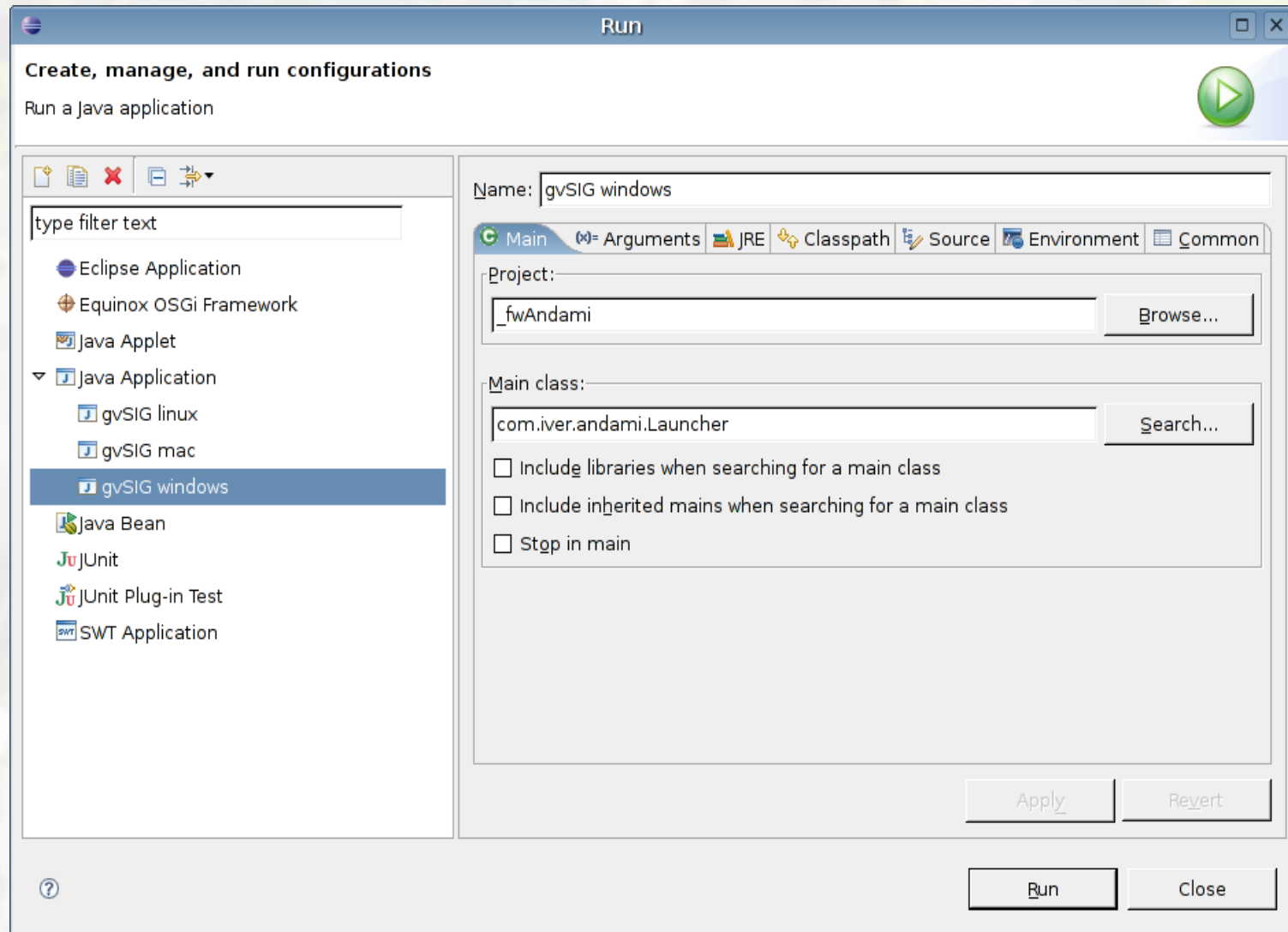
Compilación de gvSIG (automática)



Compilación de gvSIG (automática)



Ejecución de gvSIG



Depuración

Ejecución un poco especial

Sirve para entender mejor el programa, o para encontrar errores

Permite poner puntos de parada en el código, y hacer ejecuciones paso a paso

De esta forma se pueden ver o alterar los resultados intermedios

Navegación por el código. Teclas rápidas (I)

F3 nos lleva a la definición de un método o clase

CTRL+MAY+O nos añade *imports* faltantes

F4 nos muestra la jerarquía de herencias de una clase

CTRL+I indenta la(s) línea(s) automáticamente

Navegación por el código. Teclas rápidas (II)

En modo depuración:

F6 avanza una línea

F5 avanza una línea entrando dentro de la próxima función a ejecutar

F7 avanza hasta el punto en el que se ha llamado a la función actual

CTRL+MAY+I evalúa la expresión seleccionada y muestra el resultado

Modelo de Objetos (Andami)

Andami: Framework que permite la construcción de aplicaciones MDI extensibles a base de un mecanismo de plugins

Clases más importantes:

Launcher => Arranca la aplicación e inicializa todos los plugins

PluginServices => Clase base para obtener todos los servicios ofertados a los plugins plugins. Tiene métodos para obtener MainFrame, MDIManager, traducir cadenas, ejecutar tareas en segundo plano, recuperar instancias de otras extensiones, etc.

IExtension => Interface que deben implementar todos los plugins. Es la base sobre la que se añaden menus, botones, herramientas, y/o nuevas funcionalidades..

IAndamiView => Cada ventana añadida al framework debe implementar este interface. Define cómo se mostrará esa ventana. (Modal, resizable, iconifiable, etc).

Las extensiones se controlan desde el fichero "config.xml". El esquema de este fichero se define en el archivo "plugin-config.xsd".

Modelo de Objetos

(F M a p)

Clases más importantes:

MapContext => Contiene todas las capas e información acerca de la proyección, viewport, etc.
Dibuja, imprime y maneja algunos eventos.

MapControl => Componente que sabe cómo pintar (en un thread aparte) una instancia de MapContext. Está muy relacionado también con los tools.

ViewPort => Mantiene información acerca de las transformaciones a hacer (A fines o not. Genera los eventos de cambio de extent visible).

FLayers => Colección de capas. Puede ser jerárquica.

Packages:

Core => Interfaces básicos como IFeature e Igeometry + algunas implementaciones

Layers => Todo tipo de capas (raster, vectorial, wms...)

Drivers => Los sistemas lectores de datos.

Rendering => Leyendas y simbología

Strategies => Se definen las operaciones que se van a hacer con los datos.

Tools => Behaviors, tools y sus eventos.

Modelo de Objetos (gvSIG)

gvSIG: Plugin que convierte a Andami en un cliente GIS. Usa FMap como librería principal. La mayoría de las clases aquí tendrán que ver con el GUI (interface de usuario).

Clases más importantes:

Las que están en el package com.iver.cit.gvsig. Son las “extensions” a andami, las clases que implementan IExtension y aparecen en el fichero “config.xml”.

Algunos ejemplos::

AddLayer: Abre un diálogo que permite añadir capas (basadas en fichero, wms, etc)

ViewControls: Define botones como (zoom, pan, info, select, etc.

LayoutControls: La mayoría de los botones y herramientas relacionadas con el Layout.

ProjectExtension: Maneja el proyecto, y es el punto de entrada para el resto de documentos (View, Tables and Layouts).

Otras clases interesantes: View, TOC, FlegendManagerWindow, Table y Layout

3-5 Dic'08

4^{as}
Jornadas
Internacionales

gvSIG

Palacio de Congresos de Valencia



Librerías Usadas

JTS (Java Topology Suite)

Geotools2

Log4java

Batik

Castor

ErMapper

MrSID

GDAL

GDBMS

3-5 Dic'08

4^{as}
Jornadas
Internacionales

gvSIG

Palacio de Congresos de Valencia



Ejemplos

Hola Mundo

Información personalizada

Conexión a servicio remoto

Contacto

<http://www.gvsig.gva.es>

<http://www.gvsig.org>

<http://www.sextantegis.com/>

Víctor Olaya (volaya@unex.es)

Fco. José Peñarrubia (fjp@scolab.es)

César Martínez Izquierdo (cesar.martinez@iver.es)