KParts

Una introducción a KParts

Rafael Fernández López

Noviembre, 2008

Un poco de historia

XmlGui y KParts

- KParts se hizo público en la rama de KDE 2, hace unos 7 años
- XmlGui ya existía por aquel entonces (se comenzó en 1997 aproximadamente)
- Código muy estable
- Debido a grandísimos cambios en Qt, el código ha ido evolucionando, quedando algunos "restos" que todavía tienen que ser pulidos
- Mejorable a día de hoy
  - Tests automatizados → fundamentales (QTestLib)
    - *Give me a way to break it and I will find a way to fix it*
- Principal consumidor
  - Konqueror

¿Qué es XmlGui?

- Framework para desarrolladores
- Objetivos:
  - Definir interfaces gráficas de usuario mediante ficheros con estructura XML
    - Consistencia
  - Modificar "en caliente" la interfaz de una aplicación o un componente
    - Cambio de acciones en una barra de herramientas
  - Permitir cargar componentes que amplíen la interfaz de origen
    - Plugins
  - Estructura bien definida mediante ficheros DTD

Objetivos del cliente (1)

En general, hay dos tipos de clientes:

- Leer uno o varios tipos de ficheros determinados
  - Mejora la experiencia del usuario de manera muy focal
  - Se centra en un servicio determinado propocionado por un componente determinado
  - Se conoce de antemano qué tipo de fichero tratará de leerse de manera embebida
    - Importante: se conocerá en tiempo de ejecución si es posible leerlo

Ejemplo

Administrador de documentos científicos. Podría preguntar al sistema si existe algún servicio de lectura de ficheros que soporte el formato PDF

Objetivos del cliente (2)

- Leer el máximo número de tipos de ficheros posible
  - Proporcionar al usuario la máxima comodidad posible
  - Permite maximizar la multitud de tipos de ficheros distintos sin abrir una aplicación separada

Ejemplo

Konqueror. Si está configurado para ello, cuando se le solicite abrir un fichero, tratará de embeberlo si hay algún componente que sabe leerlo. En caso contrario, lanzará la aplicación encargada de ese tipo de fichero

Objetivos del cliente (3)

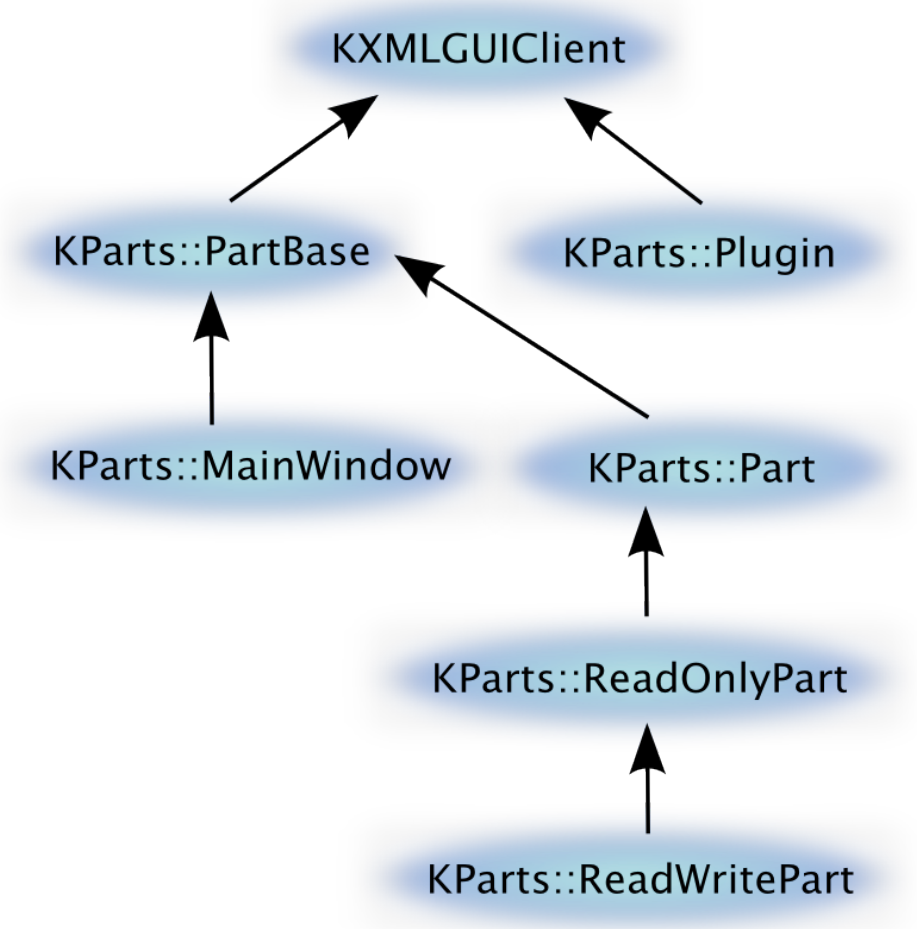
Ambos consiguen:

- Se benefician de las mejoras que se hagan en el componente automáticamente
- Mejoran la navegación
  - El componente que provee el servicio es “embebido” en la aplicación contenedora
- Evita duplicar código

Objetivos del servicio

- Ofrecer una funcionalidad muy específica e independiente del contexto
  - Añadir menús
  - Añadir acciones a una (o varias) barra(s) de herramientas
  - Ofrecer sus propias barras de herramientas
  - Éste a su vez podría ser ampliado mediante plugins
- ¿Merece la pena escribir un servicio? Depende:
  - Es incorrecta la idea de escribir un servicio exportando un widget
    - En este sentido ya contamos con las librerías de siempre
    - Merece la pena crear un servicio que exporte una funcionalidad relativamente compleja, que posiblemente, lea tipos de ficheros que no sean soportados por otras aplicaciones, aunque esto no es un requerimiento forzoso

Estructura


KXMLGUIClient
KParts::PartBase
KParts::Plugin
KParts::MainWindow
KParts::Part
KParts::ReadOnlyPart
KParts::ReadWritePart

Gracias. ¿Preguntas?

Rafael Fernández López
ereslibre@kde.org